```
; ****************************************************************************
; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3
; ----------------------------------------------------------------------------
; Last Update: 12/04/2021
; ----------------------------------------------------------------------------
; 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[CA6C]                   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[C66C]                   mov   [mem_1m_1k], cx
00000017 8916[C86C]                   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[C86C]                     mov    si, [mem_16m_64k]
00000076 8936[DF0E]                     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[5209]                       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[9242]                       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[E10E]                     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[C86C]                     mov    di, [mem_16m_64k]
000000DD 893E[DF0E]                     mov    [real_mem_16m_64k], di

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

000000E5 8936[C86C]                     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) - v2.0.0 - diskinit.s
                                    <1> ; ----------------------------------------------------------------------------
                                    <1> ; Last Update: 09/07/2016 (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> ; 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[CD6C]                 <1>       inc    byte [hdc]   ; count of hard disks (EDD present)
00000103 8816[CC6C]                 <1>         mov     [last_drv], dl  ; last hard disk number
00000107 BB[506C]                   <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[CE6C]               <1>         mov    si, fd0_type
                                <1> L3:
                                <1>         ; 14/01/2015
00000118 8816[CB6C]             <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 E8BC01                 <1>         call   set_disk_parms
                                <1>         ; 10/12/2014
00000127 81FE[CE6C]             <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[CD6C]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[CB6C]             <1>         mov    [drv], dl
00000143 8816[CC6C]             <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 0F828601               <1>         jc      L13  ; fixed disk drive not ready
0000014F 8816[CD6C]             <1>         mov     [hdc], dl ; number of drives
                                <1>         ;; 14/01/2013
                                <1>         ;;push cx
00000153 E88D01                 <1>         call   set_disk_parms
                                <1>         ;;pop  cx
                                <1>         ;
                                <1>         ;;and  cl, 3Fh       ; sectors per track (bits 0-6)
00000156 8A16[CB6C]             <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 0F855A01               <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[CB6C]             <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[CB6C]             <1>         inc    byte [drv]
000001B2 BB[506C]               <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[CD6C]               <1>         mov    al, [hdc]
```

```
000001BD FEC8                <1>       dec    al
000001BF 0F841201            <1>       jz      L13
000001C3 80FA80              <1>       cmp    dl, 80h
000001C6 0F8673FF            <1>       jna     L6
000001CA E90801              <1>       jmp     L13
                             <1> L10:
000001CD FEC2                <1>       inc    dl
                             <1>       ; 25/12/2014
000001CF 8816[CB6C]          <1>       mov    [drv], dl
000001D3 B408                <1>       mov    ah, 08h ; Return drive parameters
000001D5 CD13                <1>       int    13h    ; (conventional function)
000001D7 0F82FA00            <1>       jc      L13
                             <1>       ; 14/01/2015
000001DB 8A16[CB6C]          <1>       mov    dl, [drv]
000001DF 52                  <1>       push   dx
000001E0 51                  <1>       push   cx
000001E1 E8FF00              <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 0F82DC00            <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[D06C]          <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[CE6C]          <1>       sub    bx, hd0_type - 2 ; 15/01/2015
0000020E 81C3[1A6D]          <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 7412                <1>       jz     short L10_A0h
                             <1>                    ; 'CHS only' disks on EDD system
                             <1>                    ;  are reported with ZERO disk size
0000021C 81EB[1A6D]          <1>       sub    bx, drv.status
00000220 C1E302              <1>       shl    bx, 2
00000223 81C3[FE6C]          <1>       add    bx, drv.size ; disk size (in sectors)
00000227 8907                <1>       mov    [bx], ax
00000229 8B4412              <1>       mov    ax, [si+18]
0000022C 8907                <1>       mov    [bx], 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
0000022E BF0000              <1>       mov    di, HD0_DPT
00000231 88D0                <1>       mov    al, dl
00000233 83E003              <1>       and    ax, 3
00000236 C0E005              <1>       shl    al, 5 ; *32
00000239 01C7                <1>       add    di, ax
0000023B B80090              <1>       mov    ax, DPT_SEGM
0000023E 8EC0                <1>       mov    es, ax
                             <1>       ;
00000240 88E8                <1>       mov    al, ch ; max. cylinder number (bits 0-7)
00000242 88CC                <1>       mov    ah, cl
00000244 C0EC06              <1>       shr    ah, 6  ; max. cylinder number (bits 8-9)
00000247 40                  <1>       inc    ax     ; logical cylinders (limit 1024)
00000248 AB                  <1>       stosw
00000249 88F0                <1>       mov    al, dh ; max. head number
0000024B FEC0                <1>       inc    al
0000024D AA                  <1>       stosb         ; logical heads (limits 256)
0000024E B0A0                <1>       mov    al, 0A0h ; Indicates translated table
00000250 AA                  <1>       stosb
00000251 8A440C              <1>       mov    al, [si+12]
00000254 AA                  <1>       stosb         ; physical sectors per track
00000255 31C0                <1>       xor    ax, ax
                             <1>       ;dec   ax      ; 02/01/2015
00000257 AB                  <1>       stosw         ; precompensation (obsolete)
                             <1>       ;xor   al, al  ; 02/01/2015
00000258 AA                  <1>       stosb         ; reserved
00000259 B008                <1>       mov    al, 8   ; drive control byte
                             <1>                      ; (do not disable retries,
                             <1>                     ; more than 8 heads)
0000025B AA                  <1>       stosb
0000025C 8B4404              <1>       mov    ax, [si+4]
0000025F AB                  <1>       stosw         ; physical number of cylinders
                             <1>       ;push  ax      ; 02/01/2015
00000260 8A4408              <1>       mov    al, [si+8]
00000263 AA                  <1>       stosb         ; physical num. of heads (limit 16)
00000264 29C0                <1>       sub    ax, ax
                             <1>       ;pop   ax      ; 02/01/2015
00000266 AB                  <1>       stosw         ; landing zone (obsolete)
00000267 88C8                <1>       mov    al, cl  ; logical sectors per track (limit 63)
00000269 243F                <1>       and    al, 3Fh
0000026B AA                  <1>       stosb
                             <1>       ;sub   al, al  ; checksum
```

```
                        <1>       ;stosb
                        <1>       ;
0000026C 83C61A         <1>       add    si, 26   ; (BIOS) DPTE address pointer
0000026F AD             <1>       lodsw
00000270 50             <1>       push   ax      ; (BIOS) DPTE offset
00000271 AD             <1>       lodsw
00000272 50             <1>       push   ax      ; (BIOS) DPTE segment
                        <1>       ;
                        <1>       ; checksum calculation
00000273 89FE           <1>       mov    si, di
00000275 06             <1>       push   es
00000276 1F             <1>       pop    ds
                        <1>       ;mov   cx, 16
00000277 B90F00         <1>       mov    cx, 15
0000027A 29CE           <1>       sub    si, cx
0000027C 30E4           <1>       xor    ah, ah
                        <1>       ;del   cl
                        <1> L11:
0000027E AC             <1>       lodsb
0000027F 00C4           <1>       add    ah, al
00000281 E2FB           <1>       loop   L11
                        <1>       ;
00000283 88E0           <1>       mov    al, ah
00000285 F6D8           <1>       neg    al     ; -x+x = 0
00000287 AA             <1>       stosb         ; put checksum in byte 15 of the tbl
                        <1>       ;
00000288 1F             <1>       pop    ds     ; (BIOS) DPTE segment
00000289 5E             <1>       pop    si     ; (BIOS) DPTE offset
                        <1>       ;
                        <1>       ; 23/02/2015
0000028A 57             <1>       push   di
                        <1>       ; ES:DI points to DPTE (FDPTE) location
                        <1>       ;mov   cx, 8
0000028B B108           <1>       mov    cl, 8
0000028D 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)
0000028F 5B             <1>       pop    bx
00000290 8CC8           <1>       mov    ax, cs
00000292 8ED8           <1>       mov    ds, ax
00000294 268B07         <1>       mov    ax, [es:bx]
00000297 3DF001         <1>       cmp    ax, 1F0h
0000029A 7418           <1>       je     short L11a
0000029C 3D7001         <1>       cmp    ax, 170h
0000029F 7413           <1>       je     short L11a
                        <1>       ; invalidation
                        <1>       ; (because base port address is not 1F0h or 170h)
000002A1 30FF           <1>       xor    bh, bh
000002A3 88D3           <1>       mov    bl, dl
000002A5 80EB80         <1>       sub    bl, 80h
000002A8 C687[D06C]00   <1>       mov    byte [bx+hd0_type], 0 ; not a valid disk drive !
000002AD 808F[1C6D]F0   <1>         or      byte [bx+drv.status+2], 0F0h ; (failure sign)
000002B2 EB14           <1>       jmp    short L11b
                        <1> L11a:
                        <1>       ; LBA validation
000002B4 268A4704       <1>       mov    al, [es:bx+4] ; Head register upper nibble
000002B8 A840           <1>       test   al, 40h ; LBA bit (bit 6)
000002BA 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)
000002BC 28FF           <1>       sub    bh, bh
000002BE 88D3           <1>       mov    bl, dl
000002C0 80EB80         <1>       sub    bl, 80h ; 26/02/2015
000002C3 80A7[1C6D]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:
000002C8 3A16[CC6C]     <1>       cmp    dl, [last_drv] ; 25/12/2014
000002CC 7307           <1>       jnb    short L13
000002CE E9FCFE         <1>       jmp    L10
                        <1> L12:
                        <1>       ; Restore data registers
000002D1 8CC8           <1>       mov    ax, cs
000002D3 8ED8           <1>       mov    ds, ax
                        <1> L13:
                        <1>       ; 13/12/2014
000002D5 0E             <1>       push   cs
000002D6 07             <1>       pop    es
                        <1> L14:
000002D7 B411           <1>       mov    ah, 11h
000002D9 CD16           <1>       int    16h
000002DB 7466           <1>       jz     short L16 ; no keys in keyboard buffer
000002DD B010           <1>       mov    al, 10h
000002DF CD16           <1>       int    16h
000002E1 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
000002E3 28FF           <1>       sub    bh, bh
000002E5 8A1E[CB6C]     <1>       mov    bl, [drv]
000002E9 80FB80         <1>       cmp    bl, 80h
000002EC 7203           <1>       jb     short sdp0
000002EE 80EB7E         <1>       sub    bl, 7Eh
                        <1> sdp0:
000002F1 81C3[1A6D]     <1>       add    bx, drv.status
000002F5 C60780         <1>       mov    byte [bx], 80h ; 'Present' flag
                        <1>       ;
000002F8 88E8           <1>       mov    al, ch ; last cylinder (bits 0-7)
000002FA 88CC           <1>       mov    ah, cl ;
000002FC C0EC06         <1>       shr    ah, 6  ; last cylinder (bits 8-9)
```

```
731 000002FF 81EB[1A6D]          <1>       sub    bx, drv.status
732 00000303 D0E3                <1>       shl    bl, 1
733 00000305 81C3[D46C]          <1>       add    bx, drv.cylinders
734 00000309 40                  <1>       inc    ax  ; convert max. cyl number to cyl count
735 0000030A 8907                <1>       mov    [bx], ax
736 0000030C 50                  <1>       push   ax ; ** cylinders
737 0000030D 81EB[D46C]          <1>       sub    bx, drv.cylinders
738 00000311 81C3[E26C]          <1>       add    bx, drv.heads
739 00000315 30E4                <1>       xor    ah, ah
740 00000317 88F0                <1>       mov    al, dh ; heads
741 00000319 40                  <1>       inc    ax
742 0000031A 8907                <1>       mov    [bx], ax
743 0000031C 81EB[E26C]          <1>       sub    bx, drv.heads
744 00000320 81C3[F06C]          <1>       add    bx, drv.spt
745 00000324 30ED                <1>       xor    ch, ch
746 00000326 80E13F              <1>       and    cl, 3Fh      ; sectors (bits 0-6)
747 00000329 890F                <1>       mov    [bx], cx
748 0000032B 81EB[F06C]          <1>         sub    bx, drv.spt
749 0000032F D1E3                <1>       shl    bx, 1
750 00000331 81C3[FE6C]          <1>       add    bx, drv.size ; disk size (in sectors)
751                              <1>       ; LBA size = cylinders * heads * secpertrack
752 00000335 F7E1                <1>       mul    cx
753 00000337 89C2                <1>       mov    dx, ax ; heads*spt
754 00000339 58                  <1>       pop    ax ; ** cylinders
755 0000033A 48                  <1>       dec    ax ; 1 cylinder reserved (!?)
756 0000033B F7E2                <1>       mul    dx ; cylinders * (heads*spt)
757 0000033D 8907                <1>       mov    [bx], ax
758 0000033F 895702              <1>       mov    [bx+2], dx
759                              <1>       ;
760                              <1>       ;pop   bx
761 00000342 C3                  <1>       retn
762                              <1>
763                              <1> L16:  ; 28/05/2016
```

```
370
371                                        ; 10/11/2014
372 00000343 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 00000344 B080                          mov    al, 80h
381 00000346 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 00000348 31C0                          xor    ax, ax
392 0000034A 8EC0                          mov    es, ax ; ES = 0
393                                        ;
394                                        ; 04/07/2016 - TRDOS 386 (64K - 128K kernel)
395 0000034C 31F6                                 xor    si, si
396 0000034E 31FF                          xor    di, di
397 00000350 B90040                        mov    cx, 16384
398 00000353 F366A5                        rep    movsd
399                                        ;
400 00000356 06                            push   es ; 0
401 00000357 68[5B03]                      push   L17
402 0000035A CB                            retf
403                                    L17:
404 0000035B B90010                               mov    cx, 1000h
405 0000035E 8EC1                                 mov    es, cx  ; 1000h
406 00000360 01C9                                 add    cx, cx
407 00000362 8ED9                                 mov    ds, cx  ; 2000h
408 00000364 29F6                                 sub    si, si
409 00000366 29FF                                 sub    di, di
410 00000368 B90040                               mov    cx, 16384
411 0000036B F366A5                               rep    movsd
412
413                                               ; Turn off the floppy drive motor
414 0000036E BAF203                                 mov     dx, 3F2h
415 00000371 EE                                     out     dx, al ; 0 ; 31/12/2013
416
417                                               ; Enable access to memory above one megabyte
418                                    L18:
419 00000372 E464                                 in     al, 64h
420 00000374 A802                                 test   al, 2
421 00000376 75FA                                 jnz    short L18
422 00000378 B0D1                                 mov    al, 0D1h     ; Write output port
423 0000037A E664                                 out    64h, al
424                                    L19:
425 0000037C E464                                 in     al, 64h
426 0000037E A802                                 test   al, 2
427 00000380 75FA                                 jnz    short L19
428 00000382 B0DF                                 mov    al, 0DFh     ; Enable A20 line
429 00000384 E660                                 out    60h, al
430                                    ;L20:
431                                               ;
432                                               ; Load global descriptor table register
433
434                                               ;mov   ax, cs
435                                               ;mov   ds, ax
436
437 00000386 2E0F0116[386C]                       lgdt   [cs:gdtd]
438
439 0000038C 0F20C0                               mov    eax, cr0
440                                               ; or   eax, 1
441 0000038F 40                                   inc     ax
```

```
00000390 0F22C0                 mov     cr0, eax

                                ; Jump to 32 bit code

00000393 66                     db 66h                  ; Prefix for 32-bit
00000394 EA                     db 0EAh             ; Opcode for far jump
00000395 [9B030000]             dd StartPM          ; Offset to start, 32-bit
                                                    ; (1000h:StartPM = StartPM + 10000h)
00000399 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
0000039B 66B81000             mov ax, KDATA         ; Save data segment identifier
0000039F 8ED8                 mov ds, ax            ; Move a valid data segment into DS register
000003A1 8EC0                 mov es, ax            ; Move data segment into ES register
000003A3 8EE0                 mov fs, ax            ; Move data segment into FS register
000003A5 8EE8                 mov gs, ax            ; Move data segment into GS register
000003A7 8ED0                 mov ss, ax            ; Move data segment into SS register
000003A9 BC00000900           mov esp, 90000h       ; Move the stack pointer to 090000h

                        clear_bss: ; Clear uninitialized data area
                              ; 11/03/2015
000003AE 31C0                 xor  eax, eax ; 0
000003B0 B929630000           mov  ecx, (bss_end - bss_start)/4
                              ;shr  ecx, 2 ; bss section is already aligned for double words
000003B5 BF[06860100]         mov  edi, bss_start
000003BA F3AB                 rep  stosd

                        memory_init:
                              ; Initialize memory allocation table and page tables
                              ; 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
000003BC B108                 mov   cl, 8
000003BE BF00001000           mov   edi, MEM_ALLOC_TBL
000003C3 F3AB                 rep   stosd           ; clear Memory Allocation Table
                                                    ; for the first 1 MB memory
                              ;
000003C5 668B0D[C66C0000]     mov   cx, [mem_1m_1k]       ; Number of contiguous KB between
                                                    ; 1 and 16 MB, max. 3C00h = 15 MB.
000003CC 66C1E902             shr   cx, 2           ; convert 1 KB count to 4 KB count
000003D0 890D[F8880100]       mov   [free_pages], ecx
000003D6 668B15[C86C0000]     mov   dx, [mem_16m_64k]  ; Number of contiguous 64 KB blocks
                                                    ; between 16 MB and 4 GB.
000003DD 6609D2               or    dx, dx
000003E0 7413                 jz    short mi_0
                              ;
000003E2 6689D0               mov   ax, dx
000003E5 C1E004               shl   eax, 4          ; 64 KB -> 4 KB (page count)
000003E8 0105[F8880100]       add   [free_pages], eax
000003EE 0500100000           add   eax, 4096       ; 16 MB = 4096 pages
000003F3 EB07                 jmp   short mi_1
                        mi_0:
000003F5 6689C8               mov   ax, cx
000003F8 66050001             add   ax, 256               ; add 256 pages for the first 1 MB
                        mi_1:
000003FC A3[F4880100]         mov   [memory_size], eax ; Total available memory in pages
                                                    ; 1 alloc. tbl. bit = 1 memory page
                                                    ; 32 allocation bits = 32 mem. pages
                              ;
00000401 05FF7F0000           add   eax, 32767      ; 32768 memory pages per 1 M.A.T. page
00000406 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
00000409 66A3[08890100]       mov   [mat_size], ax        ; Memory Alloc. Table Size in pages
0000040F C1E00C               shl   eax, 12               ; 1 M.A.T. page = 4096 bytes
                              ;                     ; Max. 32 M.A.T. pages (4 GB memory)
00000412 89C3                 mov   ebx, eax        ; M.A.T. size in bytes
                              ; Set/Calculate Kernel's Page Directory Address
00000414 81C300001000         add   ebx, MEM_ALLOC_TBL
0000041A 891D[F0880100]       mov   [k_page_dir], ebx  ; Kernel's Page Directory address
                                                    ; just after the last M.A.T. page
                              ;
00000420 83E804               sub   eax, 4          ; convert M.A.T. size to offset value
00000423 A3[00890100]         mov   [last_page], eax   ; last page ofset in the M.A.T.
                              ;                     ; (allocation status search must be
                                                    ; stopped after here)
00000428 31C0                 xor   eax, eax
0000042A 48                   dec   eax             ; FFFFFFFFh (set all bits to 1)
0000042B 6651                 push  cx
0000042D C1E905               shr   ecx, 5          ; convert 1 - 16 MB page count to
                                                    ; count of 32 allocation bits
00000430 F3AB                 rep   stosd
00000432 6659                 pop   cx
00000434 40                   inc   eax             ; 0
00000435 80E11F               and   cl, 31          ; remain bits
00000438 7412                 jz    short mi_4
0000043A 8907                 mov   [edi], eax      ; reset
                        mi_2:
0000043C 0FAB07               bts   [edi], eax      ; 06/11/2014
0000043F FEC9                 dec   cl
00000441 7404                 jz    short mi_3
```

```
547 00000443 FEC0                       inc    al
548 00000445 EBF5                       jmp    short mi_2
549                                 mi_3:
550 00000447 28C0                       sub    al, al          ; 0
551 00000449 83C704                     add    edi, 4          ; 15/11/2014
552                                 mi_4:
553 0000044C 6609D2                     or     dx, dx          ; check 16M to 4G memory space
554 0000044F 7421                       jz     short mi_6      ; max. 16 MB memory, no more...
555                                     ;
556 00000451 B900021000                 mov    ecx, MEM_ALLOC_TBL + 512 ; End of first 16 MB memory
557                                     ;
558 00000456 29F9                       sub    ecx, edi        ; displacement (to end of 16 MB)
559 00000458 7406                       jz     short mi_5      ; jump if EDI points to
560                                                            ;         end of first 16 MB
561 0000045A D1E9                       shr    ecx, 1          ; convert to dword count
562 0000045C D1E9                       shr    ecx, 1          ; (shift 2 bits right)
563 0000045E F3AB                       rep    stosd           ; reset all bits for reserved pages
564                                                            ; (memory hole under 16 MB)
565                                 mi_5:
566 00000460 6689D1                     mov    cx, dx          ; count of 64 KB memory blocks
567 00000463 D1E9                       shr    ecx, 1          ; 1 alloc. dword per 128 KB memory
568 00000465 9C                         pushf                  ; 16/11/2014
569 00000466 48                         dec    eax             ; FFFFFFFFh (set all bits to 1)
570 00000467 F3AB                       rep    stosd
571 00000469 40                         inc    eax             ; 0
572 0000046A 9D                         popf                   ; 16/11/2014
573 0000046B 7305                       jnc    short mi_6
574 0000046D 6648                       dec    ax              ; eax = 0000FFFFh
575 0000046F AB                         stosd
576 00000470 6640                       inc    ax              ; 0
577                                 mi_6:
578 00000472 39DF                       cmp    edi, ebx        ; check if EDI points to
579 00000474 730A                       jnb    short mi_7      ; end of memory allocation table
580                                     ;                      ; (>= MEM_ALLOC_TBL + 4906)
581 00000476 89D9                       mov    ecx, ebx        ; end of memory allocation table
582 00000478 29F9                       sub    ecx, edi        ; convert displacement/offset
583 0000047A D1E9                       shr    ecx, 1          ; to dword count
584 0000047C D1E9                       shr    ecx, 1          ; (shift 2 bits right)
585 0000047E F3AB                       rep    stosd           ; reset all remain M.A.T. bits
586                                 mi_7:
587                                     ; Reset M.A.T. bits in M.A.T. (allocate M.A.T. pages)
588 00000480 BA00001000                 mov    edx, MEM_ALLOC_TBL
589                                     ;sub   ebx, edx        ; Mem. Alloc. Tbl. size in bytes
590                                     ;shr   ebx, 12                ; Mem. Alloc. Tbl. size in pages
591 00000485 668B0D[08890100]           mov    cx, [mat_size]         ; Mem. Alloc. Tbl. size in pages
592 0000048C 89D7                       mov    edi, edx
593 0000048E C1EF0F                     shr    edi, 15                ; convert M.A.T. address to
594                                                            ; byte offset in M.A.T.
595                                                            ; (1 M.A.T. byte points to
596                                                            ;          32768 bytes)
597                                                            ; Note: MEM_ALLOC_TBL address
598                                                            ; must be aligned on 128 KB
599                                                            ; boundary!
600 00000491 01D7                       add    edi, edx        ; points to M.A.T.'s itself
601                                     ; eax = 0
602 00000493 290D[F8880100]             sub    [free_pages], ecx ; 07/11/2014
603                                 mi_8:
604 00000499 0FB307                     btr    [edi], eax      ; clear bit 0 to bit x (1 to 31)
605                                     ;dec   bl
606 0000049C FEC9                       dec    cl
607 0000049E 7404                       jz     short mi_9
608 000004A0 FEC0                       inc    al
609 000004A2 EBF5                       jmp    short mi_8
610                                 mi_9:
611                                     ;
612                                     ; Reset Kernel's Page Dir. and Page Table bits in M.A.T.
613                                     ;            (allocate pages for system page tables)
614
615                                     ; edx = MEM_ALLOC_TBL
616 000004A4 8B0D[F4880100]             mov    ecx, [memory_size] ; memory size in pages (PTEs)
617 000004AA 81C1FF030000               add    ecx, 1023      ; round up (1024 PTEs per table)
618 000004B0 C1E90A                     shr    ecx, 10              ; convert memory page count to
619                                                            ; page table count (PDE count)
620                                     ;
621 000004B3 51                         push   ecx            ; (**) PDE count (<= 1024)
622                                     ;
623 000004B4 41                         inc    ecx            ; +1 for kernel page directory
624                                     ;
625 000004B5 290D[F8880100]             sub    [free_pages], ecx ; 07/11/2014
626                                     ;
627 000004BB 8B35[F0880100]             mov    esi, [k_page_dir] ; Kernel's Page Directory address
628 000004C1 C1EE0C                     shr    esi, 12              ; convert to page number
629                                 mi_10:
630 000004C4 89F0                       mov    eax, esi       ; allocation bit offset
631 000004C6 89C3                       mov    ebx, eax
632 000004C8 C1EB03                     shr    ebx, 3         ; convert to alloc. byte offset
633 000004CB 80E3FC                     and    bl,  0FCh      ; clear bit 0 and bit 1
634                                                            ;   to align on dword boundary
635 000004CE 83E01F                     and    eax, 31              ; set allocation bit position
636                                                            ;  (bit 0 to bit 31)
637                                     ;
638 000004D1 01D3                       add    ebx, edx       ; offset in M.A.T. + M.A.T. address
639                                     ;
640 000004D3 0FB303                     btr    [ebx], eax     ; reset relevant bit (0 to 31)
641                                     ;
642 000004D6 46                         inc    esi            ; next page table
643 000004D7 E2EB                       loop   mi_10          ; allocate next kernel page table
644                                                            ; (ecx = page table count + 1)
645                                     ;
646 000004D9 59                         pop    ecx            ; (**) PDE count (= pg. tbl. count)
647                                     ;
648                                     ; Initialize Kernel Page Directory and Kernel Page Tables
649                                     ;
650                                     ; Initialize Kernel's Page Directory
651 000004DA 8B3D[F0880100]             mov    edi, [k_page_dir]
```

```
000004E0 89F8                           mov    eax, edi
000004E2 0C03                           or     al, PDE_A_PRESENT + PDE_A_WRITE
                                                           ; supervisor + read&write + present
000004E4 89CA                           mov    edx, ecx     ; (**) PDE count (= pg. tbl. count)
                                mi_11:
000004E6 0500100000                     add    eax, 4096    ; Add page size (PGSZ)
                                                             ; EAX points to next page table
000004EB AB                             stosd
000004EC E2F8                           loop   mi_11
000004EE 29C0                           sub    eax, eax     ; Empty PDE
000004F0 66B90004                       mov    cx, 1024     ; Entry count (PGSZ/4)
000004F4 29D1                           sub    ecx, edx
000004F6 7402                           jz     short mi_12
000004F8 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
000004FA 8B0D[F4880100]                 mov    ecx, [memory_size] ; memory size in pages
00000500 89CA                           mov    edx, ecx     ; (***)
00000502 B003                           mov    al, PTE_A_PRESENT + PTE_A_WRITE
                                                           ; supervisor + read&write + present
                                mi_13:
00000504 AB                             stosd
00000505 0500100000                     add    eax, 4096
0000050A E2F8                           loop   mi_13
0000050C 6681E2FF03                     and    dx, 1023     ; (***)
00000511 740B                           jz     short mi_14
00000513 66B90004                       mov    cx, 1024
00000517 6629D1                         sub    cx, dx       ; from dx (<= 1023) to 1024
0000051A 31C0                           xor    eax, eax
0000051C F3AB                           rep    stosd        ; clear remain (empty) PTEs
                                                            ; of the last page table
                                mi_14:
                                        ;  Initialization of Kernel's Page Tables is OK, here.
                                        ;
0000051E 89F8                           mov    eax, edi     ; end of the last page table page
                                                             ; (beginging of user space pages)
00000520 C1E80F                         shr    eax, 15            ; convert to M.A.T. byte offset
00000523 24FC                           and    al, 0FCh     ; clear bit 0 and bit 1 for
                                                            ; aligning on dword boundary
00000525 A3[04890100]                   mov    [first_page], eax
0000052A A3[FC880100]                   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
                                        ;
0000052F A1[F0880100]                   mov     eax, [k_page_dir]
00000534 0F22D8                         mov    cr3, eax
00000537 0F20C0                         mov    eax, cr0
0000053A 0D00000080                     or     eax, 80000000h      ; set paging bit (bit 31)
0000053F 0F22C0                         mov    cr0, eax
                                       ;jmp    KCODE:StartPMP

00000542 EA                             db 0EAh             ; Opcode for far jump
00000543 [49050000]                     dd StartPMP         ; 32 bit offset
00000547 0800                           dw KCODE            ; kernel code segment descriptor

                                StartPMP:
                                        ; 06/11//2014
                                        ; Clear video page 0
                                        ;
                                        ; Temporary Code
                                        ;
00000549 B9E8030000                     mov    ecx, 80*25/2
0000054E BF00800B00                     mov    edi, 0B8000h
                                        ; 30/01/2016
                                        ;xor   eax, eax     ; black background, black fore color
00000553 B800070007                     mov    eax, 07000700h  ; black background, light gray fore color
00000558 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)
0000055A BE[624C0100]                   mov    esi, starting_msg
                                        ;; 14/08/2015 (kernel version message will appear
                                        ;;           when protected mode and paging is enabled)
0000055F 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:
00000564 B40A                           mov    ah, 0Ah ; Black background, light green forecolor
                               ;pkv_2:
                                        ; 20/08/2014
00000566 E8DD030000                     call   printk

                                        ; 'UNIX v7/x86' source code by Robert Nordier (1999)
```

```
                                        ; // Set IRQ offsets
                                        ;
                                        ;  Linux (v0.12) source code by Linus Torvalds (1991)
                                        ;
                                                        ;; ICW1
0000056B B011                           mov     al, 11h                 ; Initialization sequence
0000056D E620                           out     20h, al                 ;      8259A-1
                                        ; jmp   $+2
0000056F E6A0                           out     0A0h, al        ;       8259A-2
                                                        ;; ICW2
00000571 B020                           mov     al, 20h                 ; Start of hardware ints (20h)
00000573 E621                           out     21h, al                 ;      for 8259A-1
                                        ; jmp   $+2
00000575 B028                           mov     al, 28h                 ; Start of hardware ints (28h)
00000577 E6A1                           out     0A1h, al        ;       for 8259A-2
                                                        ;
00000579 B004                           mov     al, 04h                 ;; ICW3
0000057B E621                           out     21h, al                 ;      IRQ2 of 8259A-1 (master)
                                        ; jmp   $+2
0000057D B002                           mov     al, 02h                 ;      is 8259A-2 (slave)
0000057F E6A1                           out     0A1h, al        ;
                                                        ;; ICW4
00000581 B001                           mov     al, 01h                 ;
00000583 E621                           out     21h, al                 ;      8086 mode, normal EOI
                                        ; jmp   $+2
00000585 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)
00000587 BE[00490100]                   mov     esi, ilist
0000058C 8D3D[08860100]                 lea     edi, [idt]
                                        ; 26/03/2015
00000592 B930000000                     mov     ecx, 48             ; 48 hardware interrupts (INT 0 to INT 2Fh)
                                        ; 02/04/2015
00000597 BB00000800                     mov     ebx,  80000h
                                    rp_sidt1:
0000059C AD                             lodsd
0000059D 89C2                           mov     edx, eax
0000059F 66BA008E                       mov     dx, 8E00h
000005A3 6689C3                        mov     bx, ax
000005A6 89D8                           mov     eax, ebx        ; /* selector = 0x0008 = cs */
                                                                ; /* interrupt gate - dpl=0, present */
000005A8 AB                             stosd ; selector & offset bits 0-15
000005A9 89D0                           mov     eax, edx
000005AB AB                             stosd ; attributes & offset bits 16-23
000005AC 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)
000005AE B120                           mov     cl, 32        ; 32 software interrupts (INT 30h to INT 4Fh)
                                    rp_sidt2:
000005B0 AD                             lodsd
000005B1 21C0                           and     eax, eax
000005B3 7413                           jz      short rp_sidt3
000005B5 89C2                           mov     edx, eax
000005B7 66BA00EE                       mov     dx, 0EE00h      ; P=1b/DPL=11b/01110b
000005BB 6689C3                        mov     bx, ax
000005BE 89D8                           mov     eax, ebx        ; selector & offset bits 0-15
000005C0 AB                             stosd
000005C1 89D0                           mov     eax, edx
000005C3 AB                             stosd
000005C4 E2EA                           loop    rp_sidt2
000005C6 EB16                           jmp     short sidt_OK
                                    rp_sidt3:
000005C8 B8[730D0000]                   mov     eax, ignore_int
000005CD 89C2                           mov     edx, eax
000005CF 66BA00EE                       mov     dx, 0EE00h      ; P=1b/DPL=11b/01110b
000005D3 6689C3                        mov     bx, ax
000005D6 89D8                           mov     eax, ebx        ; selector & offset bits 0-15
                                    rp_sidt4:
000005D8 AB                             stosd
000005D9 92                             xchg    eax, edx
000005DA AB                             stosd
000005DB 92                             xchg    edx, eax
000005DC E2FA                           loop    rp_sidt4
                                    sidt_OK:
000005DE 0F011D[3E6C0000]               lidt    [idtd]
                                        ;
                                        ; TSS descriptor setup ; 24/03/2015
000005E5 B8[88880100]                   mov     eax, task_state_segment
000005EA 66A3[EA6B0000]                 mov     [gdt_tss0], ax
000005F0 C1C010                         rol     eax, 16
000005F3 A2[EC6B0000]                   mov     [gdt_tss1], al
000005F8 8825[EF6B0000]                 mov     [gdt_tss2], ah
000005FE 66C705[EE880100]68-            mov     word [tss.IOPB], tss_end - task_state_segment
00000606 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)
00000607 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.)
0000060B 0F00D8                         ltr    ax  ; Load task register
                                        ;
                                    esp0_set0:
                                        ; 30/07/2015
0000060E 8B0D[F4880100]                 mov    ecx, [memory_size] ; memory size in pages
00000614 C1E10C                         shl    ecx, 12 ; convert page count to byte count
00000617 81F900004000                   cmp    ecx, CORE ; beginning of user's memory space (400000h)
                                                         ; (kernel mode virtual address)
0000061D 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)
0000061F B900004000                     mov    ecx, CORE
                                    esp0_set1:
00000624 89CC                           mov    esp, ecx ; top of kernel stack (**tss.esp0**)
                                    esp0_set_ok:
                                        ; 30/07/2015 (**tss.esp0**)
00000626 8925[8C880100]                 mov    [tss.esp0], esp
0000062C 66C705[90880100]10-              mov     word [tss.ss0], KDATA
00000634 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!)
                                        ;
00000635 30C0                           xor    al, al       ; Enable all hardware interrupts!
00000637 E621                           out    21h, al             ; (IBM PC-AT compatibility)
00000639 EB00                           jmp    $+2          ; (All conventional PC-AT hardware
0000063B E6A1                           out    0A1h, al     ;  interrupts will be in use.)
                                                            ; (Even if related hardware component
                                                            ;  does not exist!)
                                        ; Enable NMI
0000063D B07F                           mov    al, 7Fh             ; Clear bit 7 to enable NMI (again)
0000063F E670                           out    70h, al
                                        ; 23/02/2015
00000641 90                             nop
00000642 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
00000644 6631DB                         xor    bx, bx
00000647 66BA0002                       mov    dx, 0200h    ; Row 2, column 0  ; 07/03/2015
0000064B E8CA1C0000                     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
00000650 803D[52090000]02               cmp    byte [vbe3], 2 ; 02h
00000657 770B                           ja     short vbe3_pmid_chk
                                        ;jb    short display_mem_info
00000659 0F82C9010000                   jb     display_mem_info ; 02/12/2020
0000065F E9FD010000                     jmp    check_boch_plex86_vbe
                                    vbe3_pmid_chk:
00000664 B9EA7F0000                     mov    ecx, 32768 - (20+2) ; 32766 - PMInfoBlockSize
00000669 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
0000066E 813E504D4944                   cmp    dword [esi], 'PMID'
                                        ;je    short display_vbios_product_name
00000674 740E                           je     short verify_pmib_chksum ; 15/11/2020
                                    ;chk_pmi_sign_next:
00000676 46                             inc    esi  ; inc si
00000677 E2F5                           loop   chk_pmi_sign

                                    not_valid_pmib:
00000679 FE0D[52090000]                 dec    byte [vbe3] ; 2 = VBE2 compatible
                                                           ; (vbe3 feature is defective in this vbios)
                                        ;jmp   short display_mem_info
                                        ; 02/12/2020
0000067F E9A4010000                     jmp    display_mem_info

                                    verify_pmib_chksum:
                                        ; 15/11/2020
00000684 31C0                           xor    eax, eax
                                        ;mov   ecx, eax
                                        ;mov   cl, 20
00000686 66B91400                       mov    cx, 20 ; 30/11/2020
0000068A 56                             push   esi
                                    pmib_sum_bytes:
0000068B AC                             lodsb
0000068C 00C4                           add    ah, al
0000068E E2FB                           loop   pmib_sum_bytes
00000690 5E                             pop    esi
00000691 08E4                           or     ah, ah
```

```
00000693 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)

00000695 C605[B7850100]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
0000069C 89F7                           mov    edi, esi

                                        ;mov   esi, [VBE3INFOBLOCK+22] ; 097E00h + 16h
                                        ;            ; OemVendorNamePtr (seg16:off16)
0000069E 8B35067E0900                   mov    esi, [VBE3INFOBLOCK+6] ; 097E00h + 06h
                                                    ; OemStringPtr (seg16:off16)
000006A4 30C0                           xor    al, al  ; eax = 0
000006A6 6696                           xchg   ax, si ; ax = offset, si = 0
000006A8 C1EE0C                         shr    esi, 12 ; (to convert segment to base addr)
000006AB 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
000006AE B40C                           mov    ah, 0Ch ; Black background, light red forecolor

000006B0 E8463B0000                     call   print_kmsg

                                        ;mov   ah, 07h

000006B5 BE[99850100]                   mov    esi, vesa_vbe3_bios_msg
                                        ;call  print_kmsg
000006BA E8423B0000                     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
000006BF 31C0                           xor    eax, eax
000006C1 B900000100                     mov    ecx, 65536
000006C6 E8865C0000                     call   allocate_memory_block
                                        ;jc    short di4
000006CB 0F8250010000                   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)

000006D1 A3[20120300]                   mov    [vbe3bios_addr], eax

                                        ; set [pmid_addr] to the new location
000006D6 BE00000C00                     mov    esi, 0C0000h

                                        ; 30/11/2020
000006DB 29F7                           sub    edi, esi ; izolate offset
000006DD 01C7                           add    edi, eax ; new address
000006DF 893D[24120300]                 mov    [pmid_addr], edi ; new 'PMID' location

                                        ; Move VIDEO BIOS from 0C0000h to EAX
000006E5 B900400000                     mov    ecx, 65536/4
000006EA 89C7                           mov    edi, eax ; 30/11/2020
000006EC F3A5                           rep    movsd
```

```
1070                                    ; 02/12/2020
1071                                    ; 30/11/2020
1072                                    ; set vbe3 segment selectors
1073
1074                                    ; VBE3CS (VESA VBE3 video bios code segment)
1075 000006EE BF[F26B0000]             mov   edi, _vbe3_CS+2 ; base address bits 0..15
1076 000006F3 66AB                     stosw  ; edi = _vbe3_CS+4
1077 000006F5 C1C810                   ror   eax, 16
1078 000006F8 8807                     mov   [edi], al ; base address, bits 16..23
1079
1080                                    ; VBE3DS ('CodeSegSel' in PMInfoBlock)
1081 000006FA BF[1C6C0000]             mov   edi, _vbe3_DS+4 ; base addr bits 16..23
1082 000006FF 8807                     mov   [edi], al
1083 00000701 C1C010                   rol   eax, 16
1084 00000704 668947FE                 mov   [edi-2], ax ; base address, bits 0..15
1085
1086                                    ; VBE3BDS (BIOSDataSel in PMInfoBlock)
1087 00000708 BF[FA6B0000]             mov   edi, _vbe3_BDS+2 ; base addr bits 0..15
1088 0000070D B800700900               mov   eax, VBE3BIOSDATABLOCK ; 1536 bytes
1089 00000712 66AB                     stosw  ; edi = _vbe3_BDS+4
1090 00000714 C1E810                   shr   eax, 16
1091 00000717 8807                     mov   [edi], al ; base address, bits 16..23
1092
1093                                    ; VBE3SS (1024 bytes)
1094 00000719 BF[226C0000]             mov   edi, _vbe3_SS+2 ; base addr bits 0..15
1095 0000071E B800600900               mov   eax, VBE3STACKADDR ; size = 1024 bytes
1096 00000723 66AB                     stosw  ; edi = _vbe3_SS+4
1097 00000725 C1E810                   shr   eax, 16
1098 00000728 8807                     mov   [edi], al ; base address, bits 16..23
1099
1100                                    ; stack pointer (esp) will be set to 1020
1101                                    ; (before VBE3 PMI call)
1102
1103                                    ; VBE3ES (max: 2048 bytes)
1104 0000072A BF[2A6C0000]             mov   edi, _vbe3_ES+2 ; base addr bits 0..15
1105 0000072F B800760900               mov   eax, VBE3SAVERESTOREBLOCK
1106 00000734 66AB                     stosw  ; edi = _vbe3_ES+4
1107 00000736 C1E810                   shr   eax, 16
1108 00000739 8807                     mov   [edi], al ; base address, bits 16..23
1109
1110                                    ;Note: low word of _VBE3_ES base address will be
1111                                    ;      set -again- by VBE3 PMI caller routine
1112
1113                                    ; 09/12/2020
1114                                    ;; set pmi32 (as VBE3 PMI is ready)
1115                                    ;inc  byte [pmi32] ; = 1
1116
1117                                    ; KCODE16 (set PMI far return segment)
1118 0000073B BF[326C0000]             mov   edi, _16bit_CS+2 ; base addr bits 0..15
1119 00000740 B8[C8070000]             mov   eax, pminit_return_addr16
1120 00000745 66AB                     stosw  ; edi = _16bit_CS+4
1121 00000747 C1E810                   shr   eax, 16
1122 0000074A 8807                     mov   [edi], al ; base address, bits 16..23
1123
1124                                    ; 30/11/2020
1125                                    ; clear mem from VBE3 BIOS data area emu block
1126                                    ; to end of vbe3 buffers
1127
1128                                    ; 01/12/2020
1129 0000074C BF00700900               mov   edi, VBE3BIOSDATABLOCK ; 97000h
1130                                    ;mov  cx, (VBE3INFOBLOCK-VBE3BIOSDATABLOCK)/4
1131                                    ;     ; ecx = 3584/4 double words
1132                                    ; 21/12/2020
1133 00000751 66B90003                 mov   cx, (VBE3MODEINFOBLOCK-VBE3BIOSDATABLOCK)/4
1134                                          ; ecx = 3072/4 double words
1135                                    ;xor  eax, eax
1136 00000755 30C0                     xor   al, al
1137 00000757 F3AB                     rep   stosd
1138
1139                                    ; Filling PMInfoBlock selector fields
1140 00000759 8B3D[24120300]           mov   edi, [pmid_addr]
1141 0000075F 66C747083800             mov   word [edi+PMInfo.BIOSDataSel], VBE3BDS
1142 00000765 66C7470A4000                   mov   word [edi+PMInfo.A0000Sel], VBE3A000
1143 0000076B 66C7470C4800                   mov   word [edi+PMInfo.B0000Sel], VBE3B000
1144 00000771 66C7470E5000                   mov   word [edi+PMInfo.B8000Sel], VBE3B800
1145 00000777 66C747105800                   mov   word [edi+PMInfo.CodeSegSel], VBE3DS
1146 0000077D C6471201                       mov   byte [edi+PMInfo.InProtectMode], 1
1147
1148                                         ; Calculate and write checksum byte
1149 00000781 89FE                           mov   esi, edi
1150 00000783 B113                           mov   cl, PMInfo.size - 1
1151                                         ;xor  ah, ah
1152                                   pmid_chksum:
1153 00000785 AC                             lodsb
1154 00000786 00C4                           add   ah, al
1155 00000788 E2FB                           loop  pmid_chksum
1156 0000078A F6DC                           neg   ah ; 1 -> 255, 255 -> 1
1157 0000078C 8826                           mov   byte [esi], ah  ; checksum
1158
1159                                         ; far call PM initialization
1160                                         ; (VBE3 video bios will return via 'retf')
1161
1162 0000078E 668B4706                       mov   ax, [edi+PMInfo.PMInitialize]
1163                                         ; 30/11/2020
1164 00000792 C1E010                         shl   eax, 16 ; save entry address in hw
1165                                         ; ax = 0
1166
1167                                         ; 02/12/2020
1168 00000795 68[EC070000]                   push  pminit_ok ; normal, near return address
1169
1170                                         ; 30/11/2020
1171                                   _VBE3PMI_fcall:
1172                                         ; ax = function, hw of eax = entry address
1173 0000079A 9C                             pushf  ; save 32 bit flags
1174 0000079B 56                             push  esi ; *
```

```
1175 0000079C 55                        push   ebp ; **
1176
1177 0000079D 89E5                      mov    ebp, esp ; save 32 bit stack pointer
1178
1179 0000079F 89C6                      mov    esi, eax
1180
1181 000007A1 FA                        cli
1182
1183                                    ; Disable interrupts (clear interrupt flag)
1184                                    ; Reset Interrupt MASK Registers (Master&Slave)
1185 000007A2 B0FF                      mov    al, 0FFh     ; mask off all interrupts
1186 000007A4 E621                      out    21h, al            ; on master PIC (8259)
1187 000007A6 EB00                      jmp    $+2  ; (delay)
1188 000007A8 E6A1                      out    0A1h, al     ; on slave PIC (8259)
1189
1190                                    ; 02/12/2020
1191 000007AA 66B86000                  mov    ax, VBE3SS
1192 000007AE 8ED0                      mov    ss, ax
1193
1194 000007B0 BCFC030000                mov    esp, 1020 ; 30/11/2020
1195
1196                                    ; 01/12/2020
1197                                    ;lss   esp, [stack16]
1198
1199 000007B5 C1E810                    shr    eax, 16      ; now, entry address is in lw
1200
1201                                    ; 30/11/2020 - 16 bit pm selector test (OK)
1202                                    ; (32 bit stack push/pop & retf with 32 bit code segment)
1203                                    ; (16 bit stack push/pop with 16 bit code segment)
1204
1205                                    ; return
1206                                    ;push  KCODE16
1207                                    ;push  0 ; 30/11/2020 (pminit_return_addr16)
1208
1209                                    ; 30/11/2020 (16 bit stack during retf from video bios)
1210 000007B8 C7042400007000              mov    dword [esp], KCODE16 << 16
1211                                                        ; ip = 0, cs = KCODE16
1212                                    ; 01/12/2020
1213                                    ;mov   dword [VBE3STACKADDR+1020], KCODE16*65536
1214
1215                                    ;mov   [jumpfar16], eax
1216
1217                                    ; 02/12/2020
1218                                    ; 30/11/2020 (32 bit stack during retf from kernel)
1219                                    ; far jump/call via retf
1220 000007BF 6A30                      push   VBE3CS ; VBE3 video bios's code segment
1221 000007C1 50                        push   eax ; PMInitialize or EntryPoint
1222
1223 000007C2 6689F0                    mov    ax, si ; restore function
1224
1225                                    ; 02/12/2020
1226 000007C5 31F6                      xor    esi, esi ; (not necessary, it is not used)
1227
1228 000007C7 CB                        retf   ; far return (to 16 bit code segment)
1229
1230                                    ; 01/12/2020
1231                                    ;db    0EAh  ; far jump to 16 bit code segment
1232                                ;jumpfar16:
1233                                    ;dd    0
1234                                    ;dw    VBE3CS
1235
1236                                ;stack16:
1237                                    ;dd    1020
1238                                    ;dw    VBE3SS
1239
1240                                    align 2
1241
1242                                pminit_return_addr16:
1243                                    ; 02/12/2020
1244                                    ; 30/11/2020
1245                                    ;;db   66h              ; Prefix for 32-bit
1246                                    ;db    0EAh             ; Opcode for far jump
1247                                    ;dd    pminit_return_addr32 ; 32 bit Offset
1248                                    ;dw    KCODE            ; 32 bit code segment
1249                                    ; 01/12/2020
1250 000007C8 EA[CF070000]0800          jmp    KCODE:pminit_return_addr32
1251
1252                                pminit_return_addr32:
1253                                    ; restore 32 bit kernel selectors and 32 bit stack addr
1254 000007CF BE10000000                mov    esi, KDATA
1255 000007D4 8EDE                      mov    ds, si
1256 000007D6 8EC6                      mov    es, si
1257 000007D8 8ED6                      mov    ss, si
1258 000007DA 89EC                      mov    esp, ebp  ; top of stack = iretd return addr
1259
1260 000007DC 5D                        pop    ebp ; **
1261 000007DD 5E                        pop    esi ; *
1262 000007DE 9D                        popf   ; restore 32 bit flags
1263
1264                                    ; enable interrupts
1265
1266 000007DF FA                        cli
1267
1268                                    ; 21/12/2020
1269 000007E0 50                        push   eax
1270
1271 000007E1 30C0                      xor    al, al    ; Enable all hardware interrupts!
1272 000007E3 E621                      out    21h, al         ; (IBM PC-AT compatibility)
1273 000007E5 EB00                      jmp    $+2       ; (All conventional PC-AT hardware
1274 000007E7 E6A1                      out    0A1h, al   ;  interrupts will be in use.)
1275                                                      ; (Even if related hardware component
1276                                                      ;  does not exist!)
1277 000007E9 58                        pop    eax
1278
1279 000007EA FB                        sti
```

```
                                        ; top of stack = return address
                                        ; ('pminit_ok' for PMinit)

000007EB C3                             retn

                                    pminit_ok:
                                        ; 03/12/2020
                                        ; (set [pmid_addr] to PMI entry point for next calls)
000007EC 8305[24120300]04               add    dword [pmid_addr], PMInfo.EntryPoint ; + 4

                                        ; 17/01/2021
                                        ; copy EDID data from temporary location to final address
000007F3 803D[92420000]4F               cmp    byte [edid], 4Fh
000007FA 7511                           jne    short vbe3h_chcl
000007FC B920000000                     mov    ecx, 32 ; 128 bytes, 32 dwords
00000801 BE007D0900                     mov    esi, VBE3EDIDINFOBLOCK ; 97D00h
00000806 BF[9C110300]                   mov    edi, edid_info
0000080B F3A5                           rep    movsd
                                        ; 17/01/2021
                                    vbe3h_chcl:
                                        ; 16/01/2021
                                        ;; 06/12/2020
                                        ;; Save video mode 03h regs/dac/bios state
                                        ;
                                        ;mov   ax, 4F04h ; VESA VBE Function 04h
                                        ;              ; Save/Restore State
                                        ;sub   dl, dl ; 0 = return buffer size
                                        ;mov   cx, 0Fh  ; ctrl/bios/dac/regs
                                        ;
                                        ;call  int10h_32bit_pmi
                                        ;; bx = number of 64-byte blocks to hold the state buff
                                        ;
                                        ;;mov  [vbe3stbufsize], bx
                                        ;; 16/01/2021
                                        ;or    word [vbe3stbsflags], 32768  ; set bit 15
                                        ;mov   ax, bx
                                        ;shl   ax, 6  ; * 64
                                        ;mov   [vbestatebufsize+30], ax
                                        ;
                                        ;; 06/12/2020
                                        ;; check 'vbe3stbufsize' (it must be <= 32)
                                        ;
                                        ;cmp   bx, 32
                                        ;ja    short display_mem_info ; light red forecolor
                                        ;
                                        ;; 16/01/2021
                                        ;or    byte [vbe3stbsflags], 1 ; set bit 0

                                        ; 30/11/2020
                                        ; Change VESA VBE3 BIOS text color in order to give
                                        ; "VBE3 PMI initialization has been successed" meaning

0000080D BE40810B00                     mov    esi, 0B8000h + 160*2 ; row 2
00000812 89F7                           mov    edi, esi
                                    vbe3h_chcl_next:
00000814 66AD                           lodsw
00000816 80FC0C                         cmp    ah, 0Ch      ; light red forecolor
00000819 750D                           jne    short display_mem_info
0000081B B40E                           mov    ah, 0Eh ; yellow forecolor
0000081D 66AB                           stosw
0000081F EBF3                           jmp    short vbe3h_chcl_next

                                    di0:
                                        ; 30/11/2020
                                        ; Memory allocation error !
00000821 C605[52090000]00               mov    byte [vbe3], 0 ; disable VBE3

                                    display_mem_info:
                                        ; 19/12/2020
                                        ; temporary
00000828 E8EA390000                     call   default_lfb_info
                                        ;
                                        ; 06/11/2014
0000082D E870390000                     call   memory_info
                                        ; 14/08/2015
                                        ;call getch ; 28/02/2015
                                    drv_init:
00000832 FB                             sti    ; Enable Interrupts
                                        ; 06/02/2015
00000833 8B15[D06C0000]                 mov    edx, [hd0_type] ; hd0, hd1, hd2, hd3
00000839 668B1D[CE6C0000]               mov    bx, [fd0_type] ; fd0, fd1
                                        ; 22/02/2015
00000840 6621DB                         and    bx, bx
00000843 756C                           jnz    short di1
                                        ;
00000845 09D2                           or     edx, edx
00000847 757A                           jnz    short di2
                                        ;
                                    setup_error:
00000849 BE[064C0100]                   mov    esi, setup_error_msg
                                    psem:
0000084E AC                             lodsb
0000084F 08C0                           or     al, al
                                        ;jz    short haltx ; 22/02/2015
00000851 747B                           jz     short di3
00000853 56                             push   esi
                                        ; 13/05/2016
00000854 BB07000000                     mov    ebx, 7 ; Black background,
                                                      ; light gray forecolor
                                                      ; Video page 0 (BH=0)
00000859 E8241A0000                     call   _write_tty
0000085E 5E                             pop    esi
0000085F EBED                           jmp    short psem
```

```
                                check_boch_plex86_vbe:
                                      ; 20/11/2020
                                      ; check Bochs/Plex86 VGABios VBE extension
                                      ; (check if TRDOS 386 v2 is running on emulators)
                                      ; BOCHS/QEMU/VIRTUALBOX
                                      ;
                                      ; ref: vbe_display_api.txt

                                      ; bochs/plex86 VGAbios VBE source code
                                      ;  by Jeroen Janssen (2002)
                                      ;  and Volker Ruppert (2003-2020)

00000861 29C0                         sub    eax, eax ; 0
00000863 66BACE01                     mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
00000867 66EF                         out    dx, ax ; VBE_DISPI_INDEX_ID register
                                      ;mov   ax, 0B0C0h ; VBE_DISPI_ID0
                                      ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
00000869 6642                         inc    dx
                                      ;out   dx, ax
                                      ;nop
0000086B 66ED                         in     ax, dx
0000086D 80FCB0                       cmp    ah, 0B0h
                                      ;jne   short not_boch_qemu_vbe
00000870 75B6                         jne    short display_mem_info
00000872 3CC5                         cmp    al, 0C5h ; it must be 0B0C4h or 0B0C5h ..
                                      ;ja    short not_boch_qemu_vbe
00000874 77B2                         ja     short display_mem_info
00000876 3CC0                         cmp    al, 0C0h ; 0B0C0h to 0B0C5h .. ; Qemu
                                      ;cmp   al, 0C4h ; 0BC04h or 0B0C5h is OK ; Bochs
                                      ;jb    short not_boch_qemu_vbe
00000878 72AE                         jb     short display_mem_info

                                      ; save VESA VBE2 bios (bochs/qemu) signature
                                      ; for enabling VBE2 functions in TRDOS 386 v2 kernel
0000087A A2[53090000]                 mov    [vbe2bios], al ; 0C4h or 0C5h (for BOCHS)
                                                            ; (0C0h-0C5h for QEMU)
0000087F C605[A2850100]32             mov    byte [vbe_vnumber], "2"
                                      ; 26/11/2020
                                      ; "BOCHS/QEMU/VIRTUALBOX VBE2 Video BIOS ..".
00000886 BE[84850100]                 mov    esi, vbe2_bochs_vbios ; BOCH/QEMU vbios msg
0000088B B40E                         mov    ah, 0Eh  ; Yellow font
0000088D E869390000                   call   print_kmsg

                                      ; this is not necessary ! (20/11/2020)
00000892 803D[53090000]C4             cmp    byte [vbe2bios], 0C4h
00000899 728D                         jb     display_mem_info    ; (QEMU)

                                      ; Display kernel version message if 0E9h hack port
                                      ; is enabled (bochs emulator feature)
0000089B 66BAE900                     mov    dx, 0E9h ; hack port for BOCHS
0000089F BE[DD850100]                 mov    esi, kernel_version_msg
                                kvmsg_next_char:
000008A4 AC                           lodsb
000008A5 08C0                         or     al, al
000008A7 7505                         jnz    short put_kvmsg_in_hack_port
                                not_boch_qemu_vbe:
000008A9 E97AFFFFFF                   jmp    display_mem_info
                                put_kvmsg_in_hack_port:
000008AE EE                           out    dx, al
000008AF EBF3                         jmp    short kvmsg_next_char

                                di1:
                                      ; supress 'jmp short T6'
                                      ;  (activate fdc motor control code)
000008B1 66C705[B4090000]90-          mov    word [T5], 9090h ; nop
000008B9 90
                                      ;
                                      ;mov   ax, int_0Eh  ; IRQ 6 handler
                                      ;mov   di, 0Eh*4    ; IRQ 6 vector
                                      ;stosw
                                      ;mov   ax, cs
                                      ;stosw
                                      ;; 16/02/2015
                                        ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
                                      ;
000008BA E84A480000                   CALL   DSKETTE_SETUP; Initialize Floppy Disks
                                      ;
000008BF 09D2                         or     edx, edx
000008C1 740B                           jz     short di3
                                di2:
000008C3 E880480000                   call         DISK_SETUP   ; Initialize Fixed Disks
000008C8 0F827BFFFFFF                   jc     setup_error
                                di3:
000008CE E8A3380000                   call   setup_rtc_int; 22/05/2015 (dsectrpm.s)
                                      ;
000008D3 E8C4410100                   call   display_disks ; 07/03/2015  (Temporary)
                                ;haltx:
                                      ; 14/08/2015
                                      ;call  getch ; 22/02/2015
                                      ;sti   ; Enable interrupts (for CPU)
                                ;     ; 29/01/2016
                                ;     sub    ah, ah ;  read time count
                                ;     call   int1Ah
                                ;     mov    edx, ecx ; 18.2 * seconds
                                ;md_info_msg_wait1:
                                ;     ; 29/01/2016
                                ;     mov    ah, 1
                                ;     call   int16h
                                ;     jz     short md_info_msg_wait2
                                ;     xor    ah, ah ; 0
                                ;       call    int16h
                                ;     jmp    short md_info_msg_ok
                                ;md_info_msg_wait2:
                                ;     sub    ah, ah  ; read time count
                                ;     call   int1Ah
```

```
                                ;       cmp     edx, ecx ; ; 18.2 * seconds
                                ;       jna     short md_info_msg_wait3
                                ;       xchg    edx, ecx
                                ;md_info_msg_wait3:
                                ;       sub     ecx, edx
                                ;       cmp     ecx, 127 ; 7 seconds (18.2 * 7)
                                ;       jb      short md_info_msg_wait1
                                ;md_info_msg_ok:

                                        ; 15/12/2020
                                        ; set initial values of LFB parameters

000008D8 803D[52090000]02               cmp     byte [vbe3], 2
000008DF 7214                           jb      short di4

                                        ;mov    ax, [def_LFB_addr]
                                        ;shl    eax, 16
                                        ;mov    [LFB_ADDR], eax
                                        ;mov    eax, 1024*768*3
                                        ;mov    [LFB_SIZE], eax

000008E1 BEFE7B0900                     mov     esi, VBE3MODEINFOBLOCK - 2
000008E6 66C7061801                     mov     word [esi], 0118h ; default vbe mode
                                                                  ; 1024*768, 24bpp
000008EB E812310000                     call    set_lfbinfo_table

000008F0 E8B25F0000                     call    allocate_lfb_pages_for_kernel
                                di4:
                                        ; 08/09/2016
000008F5 0F20C0                         mov     eax, cr0
000008F8 A810                           test    al, 10h  ; Bit 4, ET (Extension Type)
000008FA 7408                           jz      short sysinit
                                        ; 27/02/2017
000008FC FE05[AC950100]                 inc     byte [fpready]
                                        ; 80387 (FPU) is ready
00000902 DBE3                           fninit ; Initialize Floating-Point Unit
                                sysinit:
                                        ; 30/06/2015
00000904 E8FD690000                     call    sys_init
                                        ;
                                        ;jmp    cpu_reset ; 22/02/2015
                                hang:
                                        ; 23/02/2015
                                        ;sti                ; Enable interrupts
00000909 F4                             hlt
                                        ;
                                        ;nop
                                        ;; 03/12/2014
                                        ;; 28/08/2014
                                        ;mov    ah, 11h
                                        ;call   getc
                                        ;jz      _c8
                                        ;
                                        ; 23/02/2015
                                        ; 06/02/2015
                                        ; 07/09/2014
0000090A 31DB                           xor     ebx, ebx
0000090C 8A1D[1E890100]                 mov     bl, [ptty]    ; active_page
00000912 89DE                           mov     esi, ebx
00000914 66D1E6                         shl     si, 1
00000917 81C6[20890100]                 add     esi, ttychr
0000091D 668B06                         mov     ax, [esi]
00000920 6621C0                         and     ax, ax
                                        ;jz     short _c8
00000923 74E4                           jz      short hang
00000925 66C7060000                     mov     word [esi], 0
0000092A 80FB03                         cmp     bl, 3         ; Video page 3
                                        ;jb     short _c8
0000092D 72DA                           jb      short hang
                                        ;
                                        ; 13/05/2016
                                        ; 07/09/2014
                                nxtl:
0000092F 6653                           push    bx
00000931 66BB0E00                       mov     bx, 0Eh       ; Yellow character
                                                              ; on black background
                                                              ; bh = 0 (video page 0)
                                                              ; Retro UNIX 386 v1 - Video Mode 0
                                                              ; (PC/AT Video Mode 3 - 80x25 Alpha.)
00000935 6650                           push    ax
00000937 E846190000                     call    _write_tty
0000093C 6658                           pop     ax
0000093E 665B                           pop     bx
00000940 3C0D                           cmp     al, 0Dh             ; carriage return (enter)
                                        ;jne    short _c8
00000942 75C5                           jne     short hang
00000944 B00A                           mov     al, 0Ah             ; next line
00000946 EBE7                           jmp     short nxtl

                                ;_c8:
                                ;       ; 25/08/2014
                                ;       cli                         ; Disable interrupts
                                ;       mov     al, [scounter + 1]
                                ;       and     al, al
                                ;       jnz     hang
                                ;       call    rtc_p
                                ;       jmp      hang

                                        ; 27/08/2014
                                        ; 20/08/2014
                                printk:
                                          ;mov    edi, [scr_row]
                                pkl:
00000948 AC                             lodsb
00000949 08C0                           or      al, al
```

```
1594 0000094B 7404                      jz      short pkr
1595 0000094D 66AB                      stosw
1596 0000094F EBF7                      jmp     short pkl
1597                                pkr:
1598 00000951 C3                        retn
1599
1600                                ; 14/11/2020 (TRDOS 386 v2.0.3)
1601 00000952 00                    vbe3: db 0  ; VESA VBE version (must be 03h)
1602                                            ; for using video bios calls in protected mode
1603                                vbe2bios:
1604 00000953 B0                        db 0B0h ;
1605                                ;pmid_addr:
1606                                    ;dw 0  ; > 0 if 'PMID' sign is found
1607                                    ;      ;      ('pmid' offset addr in VGA bios seg, 0C000h)
1608                                    ;; 02/12/2020
1609                                    ;dw 0  ; 32 bit address in pmid_addr
1610
1611                                ; 28/02/2017
1612                                ; 22/01/2017
1613                                ; 15/01/2017
1614                                ; 14/01/2017
1615                                ; 02/01/2017
1616                                ; 25/12/2016
1617                                ; 19/12/2016
1618                                ; 10/12/2016 (callback)
1619                                ; 06/06/2016
1620                                ; 23/05/2016
1621                                ; 22/05/2016 - TRDOS 386 (TRDOS v2.0) Timer Event Modifications
1622                                ; 25/07/2015
1623                                ; 14/05/2015 (multi tasking -time sharing- 'clock', x_timer)
1624                                ; 17/02/2015
1625                                ; 06/02/2015 (unix386.s)
1626                                ; 11/12/2014 - 22/12/2014 (dsectrm2.s)
1627                                ;
1628                                ; IBM PC-XT Model 286 Source Code - BIOS2.ASM (06/10/85)
1629                                ;
1630                                ;-- HARDWARE INT  08 H - ( IRQ LEVEL 0 ) ----------------------------------------
1631                                ;     THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM FROM CHANNEL 0 OF        :
1632                                ;     THE 8254 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ AND THE DIVISOR        :
1633                                ;     IS 65536, RESULTING IN APPROXIMATELY 18.2 INTERRUPTS EVERY SECOND.     :
1634                                ;                                                                            :
1635                                ;     THE INTERRUPT HANDLER MAINTAINS A COUNT (40:6C) OF INTERRUPTS SINCE    :
1636                                ;     POWER ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.        :
1637                                ;     THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT (40:40)  :
1638                                ;     OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE           :
1639                                ;     DISKETTE MOTOR(s), AND RESET THE MOTOR RUNNING FLAGS.                   :
1640                                ;     THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH           :
1641                                ;     INTERRUPT 1CH AT EVERY TIME TICK.  THE USER MUST CODE A           :
1642                                ;     ROUTINE AND PLACE THE CORRECT ADDRESS IN THE VECTOR TABLE.        :
1643                                ;-------------------------------------------------------------------------------
1644                                ;
1645
1646                                timer_int:  ; IRQ 0
1647                                ;int_08h:   ; Timer
1648                                    ; 14/10/2015
1649                                    ; Here, we are simulating system call entry (for task switch)
1650                                    ; (If multitasking is enabled,
1651                                    ; 'clock' procedure may jump to 'sysrelease')
1652
1653 00000954 1E                      push  ds
1654 00000955 06                      push  es
1655 00000956 0FA0                    push  fs
1656 00000958 0FA8                    push  gs
1657
1658 0000095A 60                      pushad ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
1659 0000095B 66B91000                mov   cx, KDATA
1660 0000095F 8ED9                    mov   ds, cx
1661 00000961 8EC1                    mov   es, cx
1662 00000963 8EE1                    mov   fs, cx
1663 00000965 8EE9                    mov   gs, cx
1664
1665 00000967 0F20D9                  mov   ecx, cr3
1666 0000096A 890D[5C040300]          mov   [cr3reg], ecx ; save current cr3 register value/content
1667
1668                                   ; 14/01/2017
1669 00000970 3B0D[F0880100]          cmp   ecx, [k_page_dir]
1670 00000976 7409                    je    short T3
1671
1672 00000978 8B0D[F0880100]          mov   ecx, [k_page_dir]
1673 0000097E 0F22D9                  mov   cr3, ecx
1674                                T3:
1675                                   ;sti                       ; INTERRUPTS BACK ON
1676 00000981 66FF05[70890100]        INC   word [TIMER_LOW]    ; INCREMENT TIME
1677 00000988 7507                    JNZ   short T4            ; GO TO TEST_DAY
1678 0000098A 66FF05[72890100]        INC   word [TIMER_HIGH]   ; INCREMENT HIGH WORD OF TIME
1679                                T4:                        ; TEST_DAY
1680 00000991 66833D[72890100]18      CMP   word [TIMER_HIGH],018H    ; TEST FOR COUNT EQUALING 24 HOURS
1681 00000999 7519                    JNZ   short T5            ; GO TO DISKETTE_CTL
1682 0000099B 66813D[70890100]B0-     CMP   word [TIMER_LOW],0B0H
1682 000009A3 00
1683 000009A4 750E                    JNZ   short T5            ; GO TO DISKETTE_CTL
1684
1685                                ;-----      TIMER HAS GONE 24 HOURS
1686                                   ;;SUB  AX,AX
1687                                   ;MOV   [TIMER_HIGH],AX
1688                                   ;MOV   [TIMER_LOW],AX
1689 000009A6 29C0                    sub   eax, eax
1690 000009A8 A3[70890100]            mov   [TIMER_LH], eax
1691                                   ;
1692 000009AD C605[74890100]01        MOV   byte [TIMER_OFL],1
1693
1694                                ;-----      TEST FOR DISKETTE TIME OUT
1695
1696                                T5:
1697                                   ; 23/12/2014
```

```
1698 000009B4 EB1D                      jmp    short T6           ; will be replaced with nop, nop
1699                                                               ; (9090h) if a floppy disk
1700                                                               ; is detected.
1701                                    ;mov   al,[CS:MOTOR_COUNT]
1702 000009B6 A0[77890100]              mov    al, [MOTOR_COUNT]
1703 000009BB FEC8                      dec    al
1704                                    ;mov   [CS:MOTOR_COUNT], al      ; DECREMENT DISKETTE MOTOR CONTROL
1705 000009BD A2[77890100]              mov    [MOTOR_COUNT], al
1706                                    ;mov   [ORG_MOTOR_COUNT], al
1707 000009C2 750F                      JNZ    short T6           ; RETURN IF COUNT NOT OUT
1708 000009C4 B0F0                      mov    al,0F0h
1709                                    ;AND   [CS:MOTOR_STATUS],al      ; TURN OFF MOTOR RUNNING BITS
1710 000009C6 2005[76890100]            and    [MOTOR_STATUS], al
1711                                    ;and   [ORG_MOTOR_STATUS], al
1712 000009CC B00C                      MOV    AL,0CH             ; bit 3 = enable IRQ & DMA,
1713                                                               ; bit 2 = enable controller
1714                                                               ;      1 = normal operation
1715                                                               ;      0 = reset
1716                                                               ; bit 0, 1 = drive select
1717                                                               ; bit 4-7 = motor running bits
1718 000009CE 66BAF203                  MOV    DX,03F2H           ; FDC CTL PORT
1719 000009D2 EE                        OUT    DX,AL              ; TURN OFF THE MOTOR
1720                                T6:
1721                                    ;inc   word [CS:wait_count]      ; 22/12/2014 (byte -> word)
1722                                                               ; TIMER TICK INTERRUPT
1723                                    ;;inc  word [wait_count] ;;27/02/2015
1724                                    ;INT   1CH                 ; TRANSFER CONTROL TO A USER ROUTINE
1725                                    ;cli
1726 000009D3 E857040000                call   u_timer                   ; TRANSFER CONTROL TO A USER ROUTINE
1727                                    ; 23/05/2016
1728 000009D8 E8FF210100                call   clock               ; Multi Tasking control procedure
1729                                T7:
1730                                    ; 14/10/2015
1731 000009DD B020                      MOV    AL,EOI              ; GET END OF INTERRUPT MASK
1732 000009DF FA                        CLI                        ; DISABLE INTERRUPTS TILL STACK CLEARED
1733 000009E0 E620                      OUT    INTA00,AL           ; END OF INTERRUPT TO 8259 - 1
1734                                    ;
1735                                rtc_int_2:
1736                                    ; 26/12/2016
1737                                    ;mov   ecx, [cr3reg]
1738                                    ; 13/01/2017
1739 000009E2 803D[D4030300]00          cmp    byte [u.t_lock], 0        ; T_LOCK
1740 000009E9 7730                      ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
1741                                    ; 28/02/2017
1742                                    ; We need to exit if the user's IRQ callback service is in progress!
1743                                    ; (To prevent a conflict!)
1744 000009EB 803D[D8030300]00          cmp    byte [u.r_lock], 0  ; R_LOCK, IRQ callback service lock !
1745 000009F2 7727                      ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
1746                                    ; 15/01/2017
1747 000009F4 803D[80950100]02          cmp    byte [priority], 2
1748 000009FB 733A                      jnb    short T8  ; current process has a timer event (15/01/2017)
1749                                    ; 22/05/2016
1750 000009FD 803D[81950100]00          cmp    byte [p_change], 0 ; in 'set_run_sequence', in 'rtc_p'
1751 00000A04 7615                      jna    short timer_int_return ; 23/05/2016
1752
1753                                    ; 15/01/2017
1754
1755                                    ; present process must be changed with high priority process
1756                                    ;xor   al, al
1757 00000A06 31C0                      xor    eax, eax ; 26/12/2016
1758 00000A08 A2[81950100]              mov    [p_change], al ; 0
1759                                    ;mov   byte [priority], 2 ; 15/01/2017 (there is a timer event)
1760
1761 00000A0D 803D[5B030300]FF          cmp    byte [sysflg], 0FFh ; user or system space ?
1762 00000A14 7416                      je     short rtc_int_3     ; user space ([sysflg]= 0FFh)
1763
1764                                    ; system space, wait for 'sysret'
1765                                    ; to change running process
1766                                    ; with high priority (event) process
1767
1768 00000A16 A2[A8030300]              mov    [u.quant], al ; 0
1769
1770                                timer_int_return: ; 23/05/2016 - jump from 'rtc_int' ('rtc_int_2')
1771 00000A1B 8B0D[5C040300]                mov    ecx, [cr3reg]       ; previous value/content of cr3 register
1772 00000A21 0F22D9                        mov    cr3, ecx     ; restore cr3 register content
1773                                        ;
1774 00000A24 61                            popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
1775                                        ;
1776 00000A25 0FA9                          pop    gs
1777 00000A27 0FA1                          pop    fs
1778 00000A29 07                            pop    es
1779 00000A2A 1F                            pop    ds
1780                                        ;
1781 00000A2B CF                            iretd  ; return from interrupt
1782
1783                                rtc_int_3:
1784 00000A2C FE05[5B030300]                inc    byte [sysflg]       ; now, we are in system space
1785                                        ;
1786 00000A32 E9B5CD0000                    jmp     sysrelease ; change running process immediatelly
1787
1788                                T8:
1789                                        ; 13/01/2017 (eax -> ebx)
1790                                        ; callback checking... (19/12/2016)
1791 00000A37 31DB                          xor    ebx, ebx
1792 00000A39 871D[D0030300]                xchg   ebx, [u.tcb] ; callback address (0 = normal return)
1793 00000A3F 09DB                          or     ebx, ebx
1794 00000A41 74D8                          jz     short timer_int_return
1795
1796                                        ; Set user's callback routine as return address from this interrupt
1797                                        ; and set normal return address as return address from callback
1798                                        ; routine!!! (19/12/2016)
1799
1800                                        ; 14/01/2017
1801                                        ; 13/01/2017 - Timer Lock (T_LOCK)
1802 00000A43 FE05[D4030300]                inc    byte [u.t_lock]
```

```
1803 00000A49 8A0D[5B030300]              mov   cl, [sysflg]
1804 00000A4F 880D[D5030300]              mov   [u.t_mode], cl
1805
1806 00000A55 8B2D[8C880100]              mov   ebp, [tss.esp0] ; kernel stack address (for ring 0)
1807 00000A5B 83ED14                      sub   ebp, 20             ; eip, cs, eflags, esp, ss
1808 00000A5E 892D[5C030300]              mov   [u.sp], ebp
1809 00000A64 8925[60030300]              mov   [u.usp], esp
1810
1811                                      ;or   word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
1812
1813 00000A6A 8B44241C                    mov   eax, [esp+28] ; pushed eax
1814 00000A6E A3[64030300]                mov   [u.r0], eax
1815
1816 00000A73 E8600E0100                  call  wswap ; save user's registers & status
1817
1818                                      ; software int is in ring 0 but timer int must return to ring 3
1819                                      ; so, ring 3 return address and stack registers
1820                                      ; (eip, cs, eflags, esp, ss)
1821                                      ; must be copied to timer int return
1822                                      ; eip will be replaced by callback service routine address
1823
1824 00000A78 C605[5B030300]FF            mov   byte [sysflg], 0FFh ; user mode
1825
1826                                      ; system mode (system call)
1827                                      ;mov  ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
1828                                                        ; ESP (u), SS (UDATA)
1829
1830 00000A7F 8B4510                      mov   eax, [ebp+16]; SS (UDATA
1831 00000A82 89E6                        mov   esi, esp
1832 00000A84 50                          push  eax
1833 00000A85 50                          push  eax
1834 00000A86 89E7                        mov   edi, esp
1835 00000A88 893D[60030300]              mov   [u.usp], edi
1836 00000A8E B908000000                  mov   ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
1837 00000A93 F3A5                        rep   movsd
1838 00000A95 B104                        mov   cl, 4
1839 00000A97 F3AB                        rep   stosd
1840 00000A99 893D[5C030300]              mov   [u.sp], edi
1841 00000A9F 89EE                        mov   esi, ebp
1842 00000AA1 B105                        mov   cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
1843 00000AA3 F3A5                        rep   movsd
1844
1845 00000AA5 8B0D[B8030300]              mov   ecx, [u.pgdir]
1846 00000AAB 890D[5C040300]              mov   [cr3reg], ecx
1847
1848                                      ; 13/01/207 (eax -> ebx)
1849                                      ; EBX = callback routine address (virtual, not physical address!)
1850
1851                                      ; 09/01/2017
1852                                      ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'sysrele'
1853                                      ;     system call !!!
1854                                      ; 25/12/2016
1855                                      ; Callback Note: (19/12/2016)
1856                                      ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'RETN' !!!
1857                                      ;     pushf ; save flags
1858                                      ;     <callback service code>
1859                                      ;     popf  ; restore flags
1860                                      ;     retn ; return to normal running address
1861                                      ;
1862
1863                                      ; 15/01/2017
1864                                      ; 14/01/2017
1865                                      ; 13/01/2017 (eax -> ebx)
1866                                      ; 10/01/2017
1867                                  set_callback_addr:
1868                                      ; 09/01/2017 (**)
1869                                      ; 02/01/2017 (*)
1870                                      ; 25/12/2016 (*)
1871                                      ; 19/12/2016 (TRDOS 386 feature only!)
1872                                      ;
1873                                      ; This routine sets return address
1874                                      ; to start of user's interrupt
1875                                     ; service (callback) address
1876                                     ;; and sets callback 'retn' address to normal
1877                                     ;; return address of user's running code!
1878                                     ;
1879                                     ; INPUT:
1880                                     ;     EBX = callback routine/service address
1881                                     ;           (virtual, not physical address!)
1882                                     ;     [u.sp] = kernel stack, points to
1883                                     ;            user's EIP,CS,EFLAGS,ESP,SS
1884                                     ;            registers.
1885                                     ; OUTPUT:
1886                                     ;     EIP (user) = callback (service) address
1887                                     ;     CS (user) = UCODE
1888                                     ;     EFLAGS (user) = flags before callback
1889                                     ;      ESP (user) = ESP-4 (user, before callback)
1890                                     ;     [ESP](user) = EIP (user) before callback
1891                                     ;
1892                                     ; Note: If CPU was in user mode while entering
1893                                     ;     the timer interrupt service routine,
1894                                     ;     'IRET' will get return to callback routine
1895                                     ;     immediately. If CPU was in system/kernel mode
1896                                     ;     'iret' will get return to system call and
1897                                     ;     then, callback routine will be return address
1898                                     ;     from system call. (User's callback/service code
1899                                     ;     will be able to return to normal return address
1900                                     ;     via an 'retn' at the end.)
1901                                     ;
1902                                     ; Note(**): User's callback service code must be ended
1903                                     ;     with a 'sysrele' sytstem call ! (09/01/2017)
1904                                     ;
1905                                     ;     For example:
1906                                     ;
1907                                     ;     timer_callback:
```

```
1908                                    ;          ...
1909                                    ;          inc      dword [time_counter]
1910                                    ;          ...
1911                                    ;          mov eax, 39 ; 'sysrele'
1912                                    ;          int 40h ; TRDOS 386 system call (interrupt)
1913                                    ;
1914                                    ;
1915                                    ;; Note(*): User's callback service code must preserve cpu
1916                                    ;;     flags if it has any instructions which changes
1917                                    ;;     flags in the service code. (25/12/2016)
1918                                    ;;
1919                                    ;;     For example:
1920                                    ;;
1921                                    ;;     timer_callback:
1922                                    ;;         pushf ; save flags
1923                                    ;;         ; this instruction changes zero flag
1924                                    ;;         inc      dword [time_counter]
1925                                    ;;         popf ; restore flags
1926                                    ;;         retn ; return to normal user code
1927                                    ;;              (which is interrupted by the
1928                                    ;;               timer interput)
1929                                    ;;
1930
1931                                    ; 15/01/2017
1932 00000AB1 8B2D[5C030300]            mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
1933 00000AB7 895D00                   mov    [ebp], ebx
1934 00000ABA E95CFFFFFF               jmp    timer_int_return
1935
1936                                    ; 15/01/2017
1937                                    ; 13/01/2017
1938                                    ; 19/12/2016
1939                                    ; 06/06/2016
1940                                    ; 23/05/2016
1941                                    ; 22/05/2016
1942                                    ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
1943                                    ; 26/02/2015
1944                                    ; 07/09/2014
1945                                    ; 25/08/2014
1946                                rtc_int:       ; Real Time Clock Interrupt (IRQ 8)
1947                                    ; 22/05/2016
1948 00000ABF 1E                       push   ds ; ** ; 23/05/2016
1949 00000AC0 50                       push   eax ; *
1950 00000AC1 66B81000                 mov    ax, KDATA
1951 00000AC5 8ED8                     mov    ds, ax
1952                                    ;
1953 00000AC7 8A25[6E890100]           mov    ah, [RTC_2Hz] ;  2 Hz interrupt to 1 Hz function
1954 00000ACD 80F401                   xor    ah, 1
1955 00000AD0 8825[6E890100]           mov    [RTC_2Hz], ah ; 1 = 0.5 second, 0 = 1 second
1956 00000AD6 753B                     jnz    short rtc_int_return ; half second
1957                                    ; 1 second
1958                                rtc_int_0:
1959                                    ; 22/05/2016
1960 00000AD8 58                       pop    eax ; *
1961                                    ;
1962                                    ; 14/10/2015 ('timer_int')
1963                                    ; Here, we are simulating system call entry (for task switch)
1964                                    ; (If multitasking is enabled,
1965                                    ; 'clock' procedure may jump to 'sysrelease')
1966                                    ;push  ds ; ** ; 23/05/2016
1967 00000AD9 06                       push   es
1968 00000ADA 0FA0                     push   fs
1969 00000ADC 0FA8                     push   gs
1970 00000ADE 60                       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
1971 00000ADF 66B91000                 mov   cx, KDATA
1972                                    ;mov  ds, cx ; 06/06/2016
1973 00000AE3 8EC1                     mov   es, cx
1974 00000AE5 8EE1                     mov   fs, cx
1975 00000AE7 8EE9                     mov   gs, cx
1976                                    ;
1977 00000AE9 0F20D9                   mov    ecx, cr3
1978 00000AEC 890D[5C040300]           mov    [cr3reg], ecx ; save current cr3 register value/content
1979                                    ;
1980 00000AF2 803D[D4030300]00         cmp    byte [u.t_lock], 0 ; timer lock (callback) status ?
1981 00000AF9 7711                     ja     short rtc_int_1        ; yes
1982
1983                                    ; 15/01/2017
1984 00000AFB 3B0D[F0880100]           cmp    ecx, [k_page_dir]
1985 00000B01 7409                     je     short rtc_int_1
1986
1987 00000B03 8B0D[F0880100]           mov    ecx, [k_page_dir]
1988 00000B09 0F22D9                   mov    cr3, ecx
1989                                rtc_int_1:
1990                                    ; Timer event (kernel) functions must be performed with
1991                                    ; 1 second intervals - TRDOS 386 (TRDOS v2.0) feature ! -
1992                                    ;
1993                                    ; 25/08/2014
1994 00000B0C E81A030000               call   rtc_p  ; 19/05/2016 - major modification
1995
1996                                    ; 23/05/2016
1997 00000B11 28E4                     sub    ah, ah ; 0
1998                                    ; 22/05/2016 - TRDOS 386 timer event modifications
1999                                rtc_int_return: ; 19/05/2016
2000                                    ; 22/02/2015 - dsectpm.s
2001                                    ; [ source: http://wiki.osdev.org/RTC ]
2002                                    ; read status register C to complete procedure
2003                                    ;(it is needed to get a next IRQ 8)
2004 00000B13 B00C                     mov    al, 0Ch ;
2005 00000B15 E670                     out    70h, al ; select register C
2006 00000B17 90                       nop
2007 00000B18 E471                     in     al, 71h ; just throw away contents
2008                                    ; 22/02/2015
2009 00000B1A B020                     MOV    AL,EOI       ; END OF INTERRUPT
2010                                    ;CLI                ; DISABLE INTERRUPTS TILL STACK CLEARED
2011 00000B1C E6A0                     OUT    INTB00,AL    ; FOR CONTROLLER #2
2012
```

```
                                        ; 23/05/2016
00000B1E B020                           MOV    AL,EOI       ; GET END OF INTERRUPT MASK
00000B20 FA                             CLI                 ; DISABLE INTERRUPTS TILL STACK CLEARED
00000B21 E620                           OUT    INTA00,AL    ; END OF INTERRUPT TO 8259 - 1
                                        ;
                                        ; 23/05/2016
00000B23 20E4                           and    ah, ah
00000B25 0F84B7FEFFFF                   jz      rtc_int_2

                                        ; ah = 1 (half second)
00000B2B 58                             pop    eax ; *
00000B2C 1F                             pop    ds  ; **
00000B2D CF                             iretd

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

                                        ; 28/08/2014
                                irq0:
00000B2E 6A00                           push   dword 0
00000B30 EB48                           jmp    short which_irq
                                irq1:
00000B32 6A01                           push   dword 1
00000B34 EB44                           jmp    short which_irq
                                irq2:
00000B36 6A02                           push   dword 2
00000B38 EB40                           jmp    short which_irq
                                irq3:
                                        ; 20/11/2015
                                        ; 24/10/2015
00000B3A 2EFF15[9A2E0100]               call   dword [cs:com2_irq3]
00000B41 6A03                           push   dword 3
00000B43 EB35                           jmp    short which_irq
                                irq4:
                                        ; 20/11/2015
                                        ; 24/10/2015
00000B45 2EFF15[962E0100]               call   dword [cs:com1_irq4]
00000B4C 6A04                           push   dword 4
00000B4E EB2A                           jmp    short which_irq
                                irq5:
00000B50 6A05                           push   dword 5
00000B52 EB26                           jmp    short which_irq
                                irq6:
00000B54 6A06                           push   dword 6
00000B56 EB22                           jmp    short which_irq
                                irq7:
00000B58 6A07                           push   dword 7
00000B5A EB1E                           jmp    short which_irq
                                irq8:
00000B5C 6A08                           push   dword 8
00000B5E EB1A                           jmp    short which_irq
                                irq9:
00000B60 6A09                           push   dword 9
00000B62 EB16                           jmp    short which_irq
                                irq10:
00000B64 6A0A                           push   dword 10
00000B66 EB12                           jmp    short which_irq
                                irq11:
00000B68 6A0B                           push   dword 11
00000B6A EB0E                           jmp    short which_irq
                                irq12:
00000B6C 6A0C                           push   dword 12
00000B6E EB0A                           jmp    short which_irq
                                irq13:
00000B70 6A0D                           push   dword 13
00000B72 EB06                           jmp    short which_irq
                                irq14:
00000B74 6A0E                           push   dword 14
00000B76 EB02                           jmp    short which_irq
                                irq15:
00000B78 6A0F                           push   dword 15
                                        ;jmp   short which_irq

                                        ; 22/01/2017
                                        ; 19/10/2015
                                        ; 29/08/2014
                                        ; 21/08/2014
                                which_irq:
00000B7A 870424                         xchg   eax, [esp]  ; 28/08/2014
00000B7D 53                             push   ebx
00000B7E 56                             push   esi
00000B7F 57                             push   edi
00000B80 1E                             push   ds
00000B81 06                             push   es
                                        ;
00000B82 88C3                           mov    bl, al
                                        ;
00000B84 B810000000                     mov    eax, KDATA
00000B89 8ED8                           mov    ds, ax
00000B8B 8EC0                           mov    es, ax
                                        ; 19/10/2015
00000B8D FC                             cld
                                          ; 27/08/2014
00000B8E 8105[F8480100]A000-            add     dword [scr_row], 0A0h
00000B96 0000
                                        ;
00000B98 B417                           mov    ah, 17h ; blue (1) background,
                                                       ; light gray (7) forecolor
00000B9A 8B3D[F8480100]                 mov     edi, [scr_row]
00000BA0 B049                           mov    al, 'I'
00000BA2 66AB                           stosw
00000BA4 B052                           mov    al, 'R'
00000BA6 66AB                           stosw
00000BA8 B051                           mov    al, 'Q'
00000BAA 66AB                           stosw
00000BAC B020                           mov    al, ' '
```

```
2117 00000BAE 66AB                      stosw
2118 00000BB0 88D8                      mov   al, bl
2119 00000BB2 3C0A                      cmp   al, 10
2120 00000BB4 7208                      jb    short ii1
2121 00000BB6 B031                      mov   al, '1'
2122 00000BB8 66AB                      stosw
2123 00000BBA 88D8                      mov   al, bl
2124 00000BBC 2C0A                      sub   al, 10
2125                                ii1:
2126 00000BBE 0430                      add   al, '0'
2127 00000BC0 66AB                      stosw
2128 00000BC2 B020                      mov   al, ' '
2129 00000BC4 66AB                      stosw
2130 00000BC6 B021                      mov   al, '!'
2131 00000BC8 66AB                      stosw
2132 00000BCA B020                      mov   al, ' '
2133 00000BCC 66AB                      stosw
2134                                    ; 23/02/2015
2135 00000BCE 80FB07                    cmp   bl, 7 ; check for IRQ 8 to IRQ 15
2136 00000BD1 7604                      jna   ii2
2137                                    ; 22/01/2017
2138 00000BD3 B020                      mov   al, 20h  ; END OF INTERRUPT COMMAND TO
2139 00000BD5 E6A0                      out   0A0h, al ; the 2nd 8259
2140                                ii2:
2141 00000BD7 B020                      mov   al, 20h  ; END OF INTERRUPT COMMAND TO
2142 00000BD9 E620                      out   20h, al ; the 2nd 8259
2143 00000BDB E9CD010000                jmp    iiret
2144                                    ;
2145                                    ; 22/08/2014
2146                                    ;mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
2147                                    ;out   20h, al      ; 8259 PORT
2148                                    ;
2149                                    ;pop   es
2150                                    ;pop   ds
2151                                    ;pop   edi
2152                                    ;pop   esi
2153                                    ;pop   ebx
2154                                    ;pop   eax
2155                                    ;iret
2156
2157                                    ; 02/04/2015
2158                                    ; 25/08/2014
2159                               exc0:
2160 00000BE0 6A00                      push  dword 0
2161 00000BE2 E990000000                jmp   cpu_except
2162                               exc1:
2163 00000BE7 6A01                      push  dword 1
2164 00000BE9 E989000000                jmp   cpu_except
2165                               exc2:
2166 00000BEE 6A02                      push  dword 2
2167 00000BF0 E982000000                jmp   cpu_except
2168                               exc3:
2169 00000BF5 6A03                      push  dword 3
2170 00000BF7 EB7E                      jmp    cpu_except
2171                               exc4:
2172 00000BF9 6A04                      push  dword 4
2173 00000BFB EB7A                      jmp    cpu_except
2174                               exc5:
2175 00000BFD 6A05                      push  dword 5
2176 00000BFF EB76                      jmp   cpu_except
2177                               exc6:
2178 00000C01 6A06                      push  dword 6
2179 00000C03 EB72                      jmp   cpu_except
2180                               exc7:
2181 00000C05 6A07                      push  dword 7
2182 00000C07 EB6E                      jmp   cpu_except
2183                               exc8:
2184                                    ; [esp] = Error code
2185 00000C09 6A08                      push  dword 8
2186 00000C0B EB5C                      jmp   cpu_except_en
2187                               exc9:
2188 00000C0D 6A09                      push  dword 9
2189 00000C0F EB66                      jmp   cpu_except
2190                              exc10:
2191                                    ; [esp] = Error code
2192 00000C11 6A0A                      push  dword 10
2193 00000C13 EB54                      jmp   cpu_except_en
2194                              exc11:
2195                                    ; [esp] = Error code
2196 00000C15 6A0B                      push  dword 11
2197 00000C17 EB50                      jmp   cpu_except_en
2198                              exc12:
2199                                    ; [esp] = Error code
2200 00000C19 6A0C                      push  dword 12
2201 00000C1B EB4C                      jmp   cpu_except_en
2202                              exc13:
2203                                    ; [esp] = Error code
2204 00000C1D 6A0D                      push  dword 13
2205 00000C1F EB48                      jmp   cpu_except_en
2206                              exc14:
2207                                    ; [esp] = Error code
2208 00000C21 6A0E                      push  dword 14
2209 00000C23 EB44                      jmp    short cpu_except_en
2210                              exc15:
2211 00000C25 6A0F                      push  dword 15
2212 00000C27 EB4E                      jmp    cpu_except
2213                              exc16:
2214 00000C29 6A10                      push  dword 16
2215 00000C2B EB4A                      jmp    cpu_except
2216                              exc17:
2217                                    ; [esp] = Error code
2218 00000C2D 6A11                      push  dword 17
2219 00000C2F EB38                      jmp    short cpu_except_en
2220                              exc18:
2221 00000C31 6A12                      push  dword 18
```

```
2222 00000C33 EB42                      jmp    short cpu_except
2223                                 exc19:
2224 00000C35 6A13                      push   dword 19
2225 00000C37 EB3E                      jmp    short cpu_except
2226                                 exc20:
2227 00000C39 6A14                      push   dword 20
2228 00000C3B EB3A                      jmp    short cpu_except
2229                                 exc21:
2230 00000C3D 6A15                      push   dword 21
2231 00000C3F EB36                      jmp    short cpu_except
2232                                 exc22:
2233 00000C41 6A16                      push   dword 22
2234 00000C43 EB32                      jmp    short cpu_except
2235                                 exc23:
2236 00000C45 6A17                      push   dword 23
2237 00000C47 EB2E                      jmp    short cpu_except
2238                                 exc24:
2239 00000C49 6A18                      push   dword 24
2240 00000C4B EB2A                      jmp    short cpu_except
2241                                 exc25:
2242 00000C4D 6A19                      push   dword 25
2243 00000C4F EB26                      jmp    short cpu_except
2244                                 exc26:
2245 00000C51 6A1A                      push   dword 26
2246 00000C53 EB22                      jmp    short cpu_except
2247                                 exc27:
2248 00000C55 6A1B                      push   dword 27
2249 00000C57 EB1E                      jmp    short cpu_except
2250                                 exc28:
2251 00000C59 6A1C                      push   dword 28
2252 00000C5B EB1A                      jmp    short cpu_except
2253                                 exc29:
2254 00000C5D 6A1D                      push   dword 29
2255 00000C5F EB16                      jmp    short cpu_except
2256                                 exc30:
2257 00000C61 6A1E                      push   dword 30
2258 00000C63 EB04                      jmp    short cpu_except_en
2259                                 exc31:
2260 00000C65 6A1F                      push   dword 31
2261 00000C67 EB0E                      jmp   short cpu_except
2262
2263                                    ; 19/10/2015
2264                                    ; 19/09/2015
2265                                    ; 01/09/2015
2266                                    ; 28/08/2015
2267                                    ; 28/08/2014
2268                                 cpu_except_en:
2269 00000C69 87442404                  xchg   eax, [esp+4] ; Error code
2270 00000C6D 36A3[78050300]            mov    [ss:error_code], eax
2271 00000C73 58                        pop    eax  ; Exception number
2272 00000C74 870424                    xchg   eax, [esp]
2273                                           ; eax = eax before exception
2274                                           ; [esp] -> exception number
2275                                           ; [esp+4] -> EIP to return
2276                                    ; 22/01/2017
2277                                    ; 19/10/2015
2278                                    ; 19/09/2015
2279                                    ; 01/09/2015
2280                                    ; 28/08/2015
2281                                    ; 29/08/2014
2282                                    ; 28/08/2014
2283                                    ; 25/08/2014
2284                                    ; 21/08/2014
2285                                 cpu_except: ; CPU Exceptions
2286 00000C77 FC                        cld
2287 00000C78 870424                    xchg   eax, [esp]
2288                                           ; eax = Exception number
2289                                           ; [esp] = eax (before exception)
2290 00000C7B 53                        push   ebx
2291 00000C7C 56                        push   esi
2292 00000C7D 57                        push   edi
2293 00000C7E 1E                        push   ds
2294 00000C7F 06                        push   es
2295                                    ; 28/08/2015
2296 00000C80 66BB1000                  mov    bx, KDATA
2297 00000C84 8EDB                      mov    ds, bx
2298 00000C86 8EC3                      mov    es, bx
2299 00000C88 0F20DB                    mov    ebx, cr3
2300 00000C8B 53                        push   ebx ; (*) page directory
2301                                    ; 19/10/2015
2302 00000C8C FC                        cld
2303                                    ; 25/03/2015
2304 00000C8D 8B1D[F0880100]            mov    ebx, [k_page_dir]
2305 00000C93 0F22DB                    mov    cr3, ebx
2306                                    ; 28/08/2015
2307 00000C96 83F80E                    cmp    eax, 0Eh ; 14, PAGE FAULT
2308 00000C99 750F                      jne    short cpu_except_nfp
2309 00000C9B E8DF500000                call   page_fault_handler
2310 00000CA0 21C0                      and    eax, eax
2311 00000CA2 0F8401010000                jz   iiretp ; 01/09/2015
2312 00000CA8 B00E                      mov    al, 0Eh ; 14
2313                                 cpu_except_nfp:
2314                                    ; 23/08/2016
2315 00000CAA 803D[9A6E0000]03          cmp    byte [CRT_MODE], 3
2316 00000CB1 7409                      je     short cpu_except_mode_3
2317 00000CB3 50                        push   eax
2318 00000CB4 B003                      mov    al, 3
2319 00000CB6 E8470E0000                call   _set_mode
2320 00000CBB 58                        pop    eax
2321                                 cpu_except_mode_3:
2322                                    ; 02/04/2015
2323 00000CBC BB[09090000]              mov    ebx, hang
2324 00000CC1 875C241C                  xchg   ebx, [esp+28]
2325                                           ; EIP (points to instruction which faults)
2326                                           ; New EIP (hang)
```

```
2327 00000CC5 891D[7C050300]            mov   [FaultOffset], ebx
2328 00000CCB C744242008000000          mov   dword [esp+32], KCODE ; kernel's code segment
2329 00000CD3 814C242400020000          or    dword [esp+36], 200h ; enable interrupts (set IF)
2330                                    ;
2331 00000CDB 88C4                      mov   ah, al
2332 00000CDD 240F                      and   al, 0Fh
2333 00000CDF 3C09                      cmp   al, 9
2334 00000CE1 7602                      jna   short h1ok
2335 00000CE3 0407                      add   al, 'A'-':'
2336                                h1ok:
2337 00000CE5 C0EC04                    shr   ah, 4
2338 00000CE8 80FC09                    cmp   ah, 9
2339 00000CEB 7603                      jna   short h2ok
2340 00000CED 80C407                    add   ah, 'A'-':'
2341                                h2ok:
2342 00000CF0 86E0                      xchg  ah, al
2343 00000CF2 66053030                  add   ax, '00'
2344 00000CF6 66A3[504B0100]            mov   [excnstr], ax
2345                                    ;
2346                                    ; 29/08/2014
2347 00000CFC A1[7C050300]              mov   eax, [FaultOffset]
2348 00000D01 51                        push  ecx
2349 00000D02 52                        push  edx
2350 00000D03 89E3                      mov   ebx, esp
2351                                    ; 28/08/2015
2352 00000D05 B910000000                mov   ecx, 16       ; divisor value to convert binary number
2353                                                        ; to hexadecimal string
2354                                    ;mov   ecx, 10          ; divisor to convert
2355                                                          ; binary number to decimal string
2356                                b2d1:
2357 00000D0A 31D2                      xor   edx, edx
2358 00000D0C F7F1                      div   ecx
2359 00000D0E 6652                      push  dx
2360 00000D10 39C8                      cmp   eax, ecx
2361 00000D12 73F6                      jnb   short b2d1
2362 00000D14 BF[5B4B0100]              mov   edi, EIPstr ; EIP value
2363                                                      ; points to instruction which faults
2364                                    ; 28/08/2015
2365 00000D19 89C2                      mov   edx, eax
2366                                b2d2:
2367                                    ;add   al, '0'
2368 00000D1B 8A82[82420000]            mov   al, [edx+hexchrs]
2369 00000D21 AA                        stosb           ; write hexadecimal digit to its place
2370 00000D22 39E3                      cmp   ebx, esp
2371 00000D24 7606                      jna   short b2d3
2372 00000D26 6658                      pop   ax
2373 00000D28 88C2                      mov   dl, al
2374 00000D2A EBEF                      jmp   short b2d2
2375                                b2d3:
2376 00000D2C B068                      mov   al, 'h' ; 28/08/2015
2377 00000D2E AA                        stosb
2378 00000D2F B020                      mov   al, 20h          ; space
2379 00000D31 AA                        stosb
2380 00000D32 30C0                      xor   al, al    ; to do it an ASCIIZ string
2381 00000D34 AA                        stosb
2382                                    ;
2383 00000D35 5A                        pop   edx
2384 00000D36 59                        pop   ecx
2385                                    ;
2386 00000D37 B44F                      mov   ah, 4Fh      ; red (4) background,
2387                                                  ; white (F) forecolor
2388 00000D39 BE[404B0100]              mov   esi, exc_msg ; message offset
2389                                    ;
2390                                    ; 20/01/2017 (!cpu exception!)
2391                                    ;
2392 00000D3E 8105[F8480100]A000-         add    dword [scr_row], 0A0h
2392 00000D46 0000
2393 00000D48 8B3D[F8480100]              mov    edi, [scr_row]
2394                                    ;
2395 00000D4E C605[5B030300]00          mov   byte [sysflg], 0  ; system mode
2396 00000D55 FB                          sti
2397                                    ;
2398 00000D56 E8EDFBFFFF                call  printk
2399                                    ;
2400 00000D5B B410                      mov   ah, 10h
2401 00000D5D E881010000                call  int16h ; getc
2402                                    ;
2403 00000D62 B003                      mov   al, 3
2404 00000D64 E8990D0000                call  _set_mode
2405                                    ;
2406 00000D69 B801000000                mov   eax, 1
2407 00000D6E E9E0CB0000                jmp   sysexit ; terminate process !!!
2408
2409                                    ; 22/01/2017
2410                                    ; 18/04/2016
2411                                    ; 28/08/2015
2412                                    ; 23/02/2015
2413                                    ; 20/08/2014
2414                               ignore_int:
2415 00000D73 50                        push  eax
2416 00000D74 53                        push  ebx ; 23/02/2015
2417 00000D75 56                        push  esi
2418 00000D76 57                        push  edi
2419 00000D77 1E                        push  ds
2420 00000D78 06                        push  es
2421                                    ; 18/04/2016
2422 00000D79 66B81000                  mov   ax, KDATA
2423 00000D7D 8ED8                      mov   ds, ax
2424 00000D7F 8EC0                      mov   es, ax
2425                                    ; 28/08/2015
2426 00000D81 0F20D8                    mov   eax, cr3
2427 00000D84 50                        push  eax ; (*) page directory
2428                                    ;
2429 00000D85 B467                      mov   ah, 67h      ; brown (6) background,
2430                                                ; light gray (7) forecolor
```

```
2431 00000D87 BE[084A0100]              mov    esi, int_msg ; message offset
2432                                 piemsg:
2433                                        ; 27/08/2014
2434 00000D8C 8105[F8480100]A000-           add    dword [scr_row], 0A0h
2434 00000D94 0000
2435 00000D96 8B3D[F8480100]                mov    edi, [scr_row]
2436                                        ;
2437 00000D9C E8A7FBFFFF                    call   printk
2438                                        ;
2439                                        ; 23/02/2015
2440 00000DA1 B020                          mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2441 00000DA3 E6A0                          out    0A0h, al ; the 2nd 8259
2442                                        ; 22/08/2014
2443 00000DA5 B020                          mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
2444 00000DA7 E620                          out    20h, al      ; 8259 PORT
2445                                 iiretp:
2446                                        ; 22/01/2017
2447                                        ; 01/09/2015
2448                                        ; 28/08/2015
2449 00000DA9 58                            pop    eax ; (*) page directory
2450 00000DAA 0F22D8                        mov    cr3, eax
2451                                 iiret:
2452 00000DAD 07                            pop    es
2453 00000DAE 1F                            pop    ds
2454 00000DAF 5F                            pop    edi
2455 00000DB0 5E                            pop    esi
2456 00000DB1 5B                            pop    ebx ; 29/08/2014
2457 00000DB2 58                            pop    eax
2458 00000DB3 CF                            iretd
2459
2460                                        ; 23/05/2016
2461                                        ; 22/08/2014
2462                                        ; IBM PC/AT BIOS source code ----- 10/06/85 (bios.asm)
2463                                        ; (INT 1Ah)
2464                                        ;; Linux (v0.12) source code (main.c) by Linus Torvalds (1991)
2465                                 time_of_day:
2466 00000DB4 E8B55D0000                    call   UPD_IPR                  ; WAIT TILL UPDATE NOT IN PROGRESS
2467 00000DB9 726F                          jc       short time_of_day_retn ; 23/05/2016
2468 00000DBB B000                          mov    al, CMOS_SECONDS
2469 00000DBD E8C75D0000                    call   CMOS_READ
2470 00000DC2 A2[60890100]                  mov    [time_seconds], al
2471 00000DC7 B002                          mov    al, CMOS_MINUTES
2472 00000DC9 E8BB5D0000                    call   CMOS_READ
2473 00000DCE A2[61890100]                  mov    [time_minutes], al
2474 00000DD3 B004                          mov    al, CMOS_HOURS
2475 00000DD5 E8AF5D0000                    call   CMOS_READ
2476 00000DDA A2[62890100]                  mov     [time_hours], al
2477 00000DDF B006                          mov    al, CMOS_DAY_WEEK
2478 00000DE1 E8A35D0000                    call   CMOS_READ
2479 00000DE6 A2[63890100]                  mov    [date_wday], al
2480 00000DEB B007                          mov    al, CMOS_DAY_MONTH
2481 00000DED E8975D0000                    call   CMOS_READ
2482 00000DF2 A2[64890100]                  mov    [date_day], al
2483 00000DF7 B008                          mov    al, CMOS_MONTH
2484 00000DF9 E88B5D0000                    call   CMOS_READ
2485 00000DFE A2[65890100]                  mov    [date_month], al
2486 00000E03 B009                          mov    al, CMOS_YEAR
2487 00000E05 E87F5D0000                    call   CMOS_READ
2488 00000E0A A2[66890100]                  mov    [date_year], al
2489 00000E0F B032                          mov    al, CMOS_CENTURY
2490 00000E11 E8735D0000                    call   CMOS_READ
2491 00000E16 A2[67890100]                  mov    [date_century], al
2492                                        ;
2493 00000E1B B000                          mov    al, CMOS_SECONDS
2494 00000E1D E8675D0000                    call   CMOS_READ
2495 00000E22 3A05[60890100]                cmp    al, [time_seconds]
2496 00000E28 758A                          jne    short time_of_day
2497
2498                                 time_of_day_retn:
2499 00000E2A C3                            retn
2500
2501                                        ; 15/01/2017
2502                                        ; 10/06/2016
2503                                        ; 07/06/2016
2504                                        ; 06/06/2016
2505                                        ; 23/05/2016
2506                                 rtc_p:
2507 00000E2B B101                          mov    cl, 1 ; 15/01/2017
2508 00000E2D EB02                          jmp    short rtc_p0
2509                                 u_timer:
2510                                        ; Timer Events with 18.2 Hz Timer Ticks
2511                                        ; (and also timer events with RTC seconds)
2512 00000E2F 28C9                          sub    cl, cl ; mov cl, 0 ; 15/01/2017
2513                                 rtc_p0:
2514                                        ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
2515                                        ; Major Modification:
2516                                        ; Check and Perform Timer Events (for RTC)
2517                                        ; 25/08/2014 - 07/09/2014
2518                                        ; Retro UNIX 386 v1:
2519                                        ; Print Real Time Clock content
2520
2521                                        ; 15/01/2017
2522 00000E31 880D[80950100]                mov    byte [priority], cl ; 0 or 1 (not 2)
2523 00000E37 8A2D[83950100]                mov    ch, [timer_events]
2524 00000E3D 20ED                          and    ch, ch
2525 00000E3F 7420                          jz     short rtc_p3
2526
2527 00000E41 BE[60040300]                  mov    esi, timer_set  ; beginning address of
2528                                                               ; timer events space
2529                                 rtc_p1:
2530 00000E46 8B06                          mov    eax, [esi]
2531 00000E48 20C0                          and    al, al ; 0 = free, >0 = process no.
2532 00000E4A 7416                          jz     short rtc_p4
2533                                        ;
2534 00000E4C C1C810                        ror    eax, 16
```

```
                                        ; ah = response value, al = interrupt type
                                        ; 15/01/2017
                                        ; cl = interrupt source
                                        ;       1 = RTC, 0 = PIT
00000E4F 38C8                           cmp    al, cl
00000E51 750A                           jne    short rtc_p2 ; not as requested or undefined !
00000E53 3C01                           cmp    al, 1 ; 1 ; RTC interrupt ?
00000E55 7410                           je     short rtc_p5 ; yes, check for response
                                        ; 06/06/2016 - 18.2 Hz Timer Ticks
00000E57 836E080A                       sub    dword [esi+8], 10 ; 1 tick = 10
00000E5B 7613                           jna    short rtc_p6  ; continue for responding
                                rtc_p2:
                                        ; 15/01/2017 (cl -> ch)
                                        ; 07/06/2016
00000E5D FECD                           dec    ch    ; remain count of timer events
00000E5F 7501                           jnz    short rtc_p4
                                rtc_p3:
00000E61 C3                             retn
                                rtc_p4:
                                        ;cmp   esi, timer_set + 240 ; 15*16 (last event)
                                        ;jnb   short rtc_p3 ; end of timer event space
00000E62 83C610                         add    esi, 16 ; next timer event
00000E65 EBDF                           jmp    short rtc_p1
                                rtc_p5:
                                        ; current timer count ; 06/06/2016 (182)
00000E67 816E08B6000000                 sub    dword [esi+8], 182 ; 1 second (10*18.2)
00000E6E 77ED                           ja     short rtc_p2  ; check for the next
                                rtc_p6:
                                        ; it is the time of response!
00000E70 8B5E04                         mov    ebx, [esi+4] ; set (count limit) value
00000E73 895E08                         mov    [esi+8], ebx ; reset count down value
                                                            ; to count limit
                                        ; 19/12/2016
                                        ; 10/12/2016 - timer callback modification
00000E76 8B7E0C                         mov    edi, [esi+12] ; response (or callback) address
00000E79 807E0100                       cmp    byte [esi+1], 0 ; >0 = callback
00000E7D 762A                           jna    short rtc_p8

                                        ; timer callback !
00000E7F 0FB61E                         movzx  ebx, byte [esi] ; process number (>0)
00000E82 89D8                           mov    eax, ebx
00000E84 C0E302                         shl    bl, 2 ; *4
00000E87 89BB[0C010300]                 mov    [ebx+p.tcb-4], edi ; user's callback service addr
00000E8D 3A05[B3030300]                 cmp    al, [u.uno]
00000E93 7521                           jne    short rtc_p9
00000E95 893D[D0030300]                 mov    [u.tcb], edi
                                rtc_p7:
                                        ; 15/01/2017
00000E9B B002                           mov    al, 2
00000E9D A2[80950100]                   mov    [priority], al ; 2
                                        ; 10/01/2017
                                        ;mov   byte [u.pri], 2
00000EA2 A2[A9030300]                   mov    [u.pri], al ; 2
00000EA7 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
00000EA9 8827                           mov    [edi], ah     ; response value
                                        ;
00000EAB C1C010                         rol    eax, 16
                                        ; 15/01/2017
00000EAE 3A05[B3030300]                 cmp    al, [u.uno] ; running process ?
00000EB4 74E5                           je     short rtc_p7
                                rtc_p9:
                                        ; al = process number  ; 10/06/2016
00000EB6 B202                           mov    dl, 2 ; priority, 2 = event (high)
00000EB8 E8D31C0100                     call   set_run_sequence ; 19/05/2016
00000EBD 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:
00000EBF 6650                           push   ax
00000EC1 B00B                           mov    al, 0Bh  ; In-Service register
00000EC3 E620                           out    20h, al
00000EC5 EB00                           jmp short $+2
00000EC7 EB00                           jmp short $+2
00000EC9 E420                           in     al, 20h
00000ECB 2480                           and    al, 80h ; bit 7 (is it real IRQ 7 or fake?)
00000ECD 7404                           jz     short irq7_iret ; Fake (spurious) IRQ, do not send EOI
00000ECF B020                           mov    al, 20h ; EOI
00000ED1 E620                           out    20h, al
                                irq7_iret:
00000ED3 6658                           pop    ax
00000ED5 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)
                                        ;
00000ED6 D410                           db 0D4h,10h                     ; Undocumented inst. AAM
                                                                ; AH = AL / 10h
                                                                ; AL = AL MOD 10h
00000ED8 660D3030                       or ax,'00'                      ; Make it ASCII based

00000EDC 86E0                           xchg ah, al

00000EDE C3                             retn
```

```
                                ; 15/12/2020
                                real_mem_16m_64k:
00000EDF 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:
00000EE1 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:
00000EE3 9C                   <1>       pushfd ; 28/08/2014
00000EE4 0E                   <1>       push   cs
00000EE5 E801000000           <1>       call   KEYBOARD_IO_1 ; getc_int
00000EEA 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>       ;-------------------------------------------------------------------------:
                              <1>       ;     (AH)= 01H  SET THE ZERO FLAG TO INDICATE IF AN ASCII CHARACTER IS     :
                              <1>       ;              AVAILABLE TO BE READ FROM THE KEYBOARD BUFFER.             :
                              <1>       ;              (ZF)= 1 -- NO CODE AVAILABLE                               :
                              <1>       ;              (ZF)= 0 -- CODE IS AVAILABLE  (AX)= CHARACTER              :
                              <1>       ;              IF (ZF)= 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ IS :
                              <1>       ;              IN (AX), AND THE ENTRY REMAINS IN THE BUFFER.               :
                              <1>       ;              THIS WILL RETURN ONLY PC/PCAT KEYBOARD COMPATIBLE CODES     :
                              <1>       ;-------------------------------------------------------------------------:
                              <1>       ;     (AH)= 02H  RETURN THE CURRENT SHIFT STATUS IN AL REGISTER            :
                              <1>       ;              THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE         :
                              <1>       ;              EQUATES FOR @KB_FLAG                                       :
                              <1>       ;-------------------------------------------------------------------------:
                              <1>       ;     (AH)= 03H  SET TYPAMATIC RATE AND DELAY                            :
                              <1>       ;           (AL) = 05H                                                    :
                              <1>       ;           (BL) = TYPAMATIC RATE (BITS 5 - 7 MUST BE RESET TO 0)         :
                              <1>       ;                                                                         :
                              <1>       ;                    REGISTER     RATE      REGISTER     RATE            :
                              <1>       ;                     VALUE     SELECTED     VALUE     SELECTED          :
                              <1>       ;                    ------------------------------------------         :
                              <1>       ;                      00H        30.0        10H        7.5             :
                              <1>       ;                      01H        26.7        11H        6.7             :
```

```
2745                            <1>       ;                   02H        24.0        12H        6.0                 :
2746                            <1>       ;                   03H        21.8        13H        5.5                 :
2747                            <1>       ;                   04H        20.0        14H        5.0                 :
2748                            <1>       ;                   05H        18.5        15H        4.6                 :
2749                            <1>       ;                   06H        17.1        16H        4.3                 :
2750                            <1>       ;                   07H        16.0        17H        4.0                 :
2751                            <1>       ;                   08H        15.0        18H        3.7                 :
2752                            <1>       ;                   09H        13.3        19H        3.3                 :
2753                            <1>       ;                   0AH        12.0        1AH        3.0                 :
2754                            <1>       ;                   0BH        10.9        1BH        2.7                 :
2755                            <1>         ;                 0CH        10.0        1CH        2.5                 :
2756                            <1>       ;                   0DH         9.2        1DH        2.3                 :
2757                            <1>       ;                   0EH         8.6        1EH        2.1                 :
2758                            <1>       ;                   0FH         8.0        1FH        2.0                 :
2759                            <1>       ;                                                                        :
2760                            <1>       ;           (BH) = TYPAMATIC DELAY  (BITS 2 - 7 MUST BE RESET TO 0)          :
2761                            <1>       ;                                                                      :
2762                            <1>       ;                     REGISTER     DELAY                                  :
2763                            <1>       ;                      VALUE       VALUE                                  :
2764                            <1>       ;                     ------------------                                  :
2765                            <1>       ;                   00H        250 ms                                     :
2766                            <1>       ;                   01H        500 ms                                     :
2767                            <1>       ;                   02H        750 ms                                     :
2768                            <1>       ;                   03H       1000 ms                                     :
2769                            <1>       ;-------------------------------------------------------------------------:
2770                            <1>       ;     (AH)= 05H  PLACE ASCII CHARACTER/SCAN CODE COMBINATION IN KEYBOARD    :
2771                            <1>       ;               BUFFER AS IF STRUCK FROM KEYBOARD                         :
2772                            <1>       ;               ENTRY:  (CL) = ASCII CHARACTER                          :
2773                            <1>       ;                       (CH) = SCAN CODE                                  :
2774                            <1>       ;               EXIT:   (AH) = 00H = SUCCESSFUL OPERATION                 :
2775                            <1>       ;                       (AL) = 01H = UNSUCCESSFUL - BUFFER FULL           :
2776                            <1>       ;               FLAGS:  CARRY IF ERROR                                    :
2777                            <1>       ;-------------------------------------------------------------------------:
2778                            <1>       ;     (AH)= 10H  EXTENDED READ INTERFACE FOR THE ENHANCED KEYBOARD,         :
2779                            <1>       ;               OTHERWISE SAME AS FUNCTION AH=0                         :
2780                            <1>       ;-------------------------------------------------------------------------:
2781                            <1>       ;     (AH)= 11H  EXTENDED ASCII STATUS FOR THE ENHANCED KEYBOARD,           :
2782                            <1>       ;               OTHERWISE SAME AS FUNCTION AH=1                         :
2783                            <1>       ;-------------------------------------------------------------------------:
2784                            <1>       ;     (AH)= 12H  RETURN THE EXTENDED SHIFT STATUS IN AX REGISTER           :
2785                            <1>       ;               AL = BITS FROM KB_FLAG, AH = BITS FOR LEFT AND RIGHT      :
2786                            <1>       ;               CTL AND ALT KEYS FROM KB_FLAG_1 AND KB_FLAG_3             :
2787                            <1>       ; OUTPUT                                                                :
2788                            <1>       ;     AS NOTED ABOVE, ONLY (AX) AND FLAGS CHANGED                       :
2789                            <1>       ;     ALL REGISTERS RETAINED                                             :
2790                            <1>       ;-------------------------------------------------------------------------
2791                            <1>
2792                            <1> ; 12/04/2021 - TRDOS 386 v2.0.3 (32 bit push/pop)
2793                            <1> ; 15/01/2017
2794                            <1> ; 14/01/2017
2795                            <1> ; 02/01/2017
2796                            <1> ; 29/05/2016
2797                            <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
2798                            <1> int32h:  ; Keyboard BIOS
2799                            <1>
2800                            <1> KEYBOARD_IO_1:
2801                            <1>       ;sti                    ; INTERRUPTS BACK ON
2802                            <1>       ; 29/05/2016
2803 00000EEB 80642408BE        <1>         and     byte [esp+8], 10111110b ; clear zero flag and cary flag
2804                            <1>       ;
2805 00000EF0 1E                <1>       push   ds               ; SAVE CURRENT DS
2806 00000EF1 53                <1>       push   ebx              ; SAVE BX TEMPORARILY
2807                            <1>       ;push  ecx              ; SAVE CX TEMPORARILY
2808 00000EF2 66BB1000          <1>       mov    bx, KDATA
2809 00000EF6 8EDB              <1>       mov    ds, bx           ; PUT SEGMENT VALUE OF DATA AREA INTO DS
2810                            <1>       ; 14/01/2017
2811 00000EF8 8B1C24            <1>       mov    ebx, [esp]
2812                            <1>       ;; 15/01/2017
2813                            <1>       ; 02/01/2017
2814                            <1>       ;;mov  byte [intflg], 32h  ; keyboard interrupt
2815 00000EFB FB                <1>       sti
2816                            <1>       ;
2817 00000EFC 08E4              <1>       or     ah, ah           ; CHECK FOR (AH)= 00H
2818 00000EFE 745B              <1>       jz     short _K1        ; ASCII_READ
2819 00000F00 FECC              <1>       dec    ah                ; CHECK FOR (AH)= 01H
2820 00000F02 7474              <1>       jz    short _K2         ; ASCII_STATUS
2821 00000F04 FECC              <1>       dec    ah               ; CHECK FOR (AH)= 02H
2822 00000F06 7440              <1>       jz     short _K3        ; SHIFT STATUS
2823 00000F08 FECC              <1>       dec    ah               ; CHECK FOR (AH)= 03H
2824 00000F0A 0F8493000000      <1>       jz     _K300            ; SET TYPAMATIC RATE/DELAY
2825 00000F10 80EC02            <1>       sub    ah, 2            ; CHECK FOR (AH)= 05H
2826 00000F13 0F84BC000000      <1>       jz     _K500            ; KEYBOARD WRITE
2827                            <1> _KIO1:
2828 00000F19 80EC0B            <1>       sub    ah, 11           ; AH =  10H
2829 00000F1C 7431              <1>       jz     short _K1E       ; EXTENDED ASCII READ
2830 00000F1E FECC              <1>       dec    ah               ; CHECK FOR (AH)= 11H
2831 00000F20 7447              <1>       jz     short _K2E       ; EXTENDED_ASCII_STATUS
2832 00000F22 FECC              <1>       dec    ah               ; CHECK FOR (AH)= 12H
2833 00000F24 7404              <1>       jz     short _K3E       ; EXTENDED_SHIFT_STATUS
2834                            <1> _KIO_EXIT:
2835                            <1>       ; 02/01/2017
2836 00000F26 FA                <1>       cli
2837                            <1>       ;;mov  byte [intflg], 0 ;; 15/01/2017
2838                            <1>       ;
2839                            <1>       ;pop   ecx              ; RECOVER REGISTER
2840 00000F27 5B                <1>       pop    ebx              ; RECOVER REGISTER
2841 00000F28 1F                <1>       pop    ds               ; RECOVER SEGMENT
2842 00000F29 CF                <1>       iretd                   ; INVALID COMMAND, EXIT
2843                            <1>
2844                            <1>       ;----- SHIFT STATUS
2845                            <1> _K3E:                         ; GET THE EXTENDED SHIFT STATUS FLAGS
2846 00000F2A 8A25[666E0000]    <1>       mov    ah, [KB_FLAG_1]  ; GET SYSTEM SHIFT KEY STATUS
2847 00000F30 80E404            <1>       and    ah, SYS_SHIFT    ; MASK ALL BUT SYS KEY BIT
2848                            <1>       ;mov   cl, 5            ; SHIFT THEW SYSTEMKEY BIT OVER TO
2849                            <1>       ;shl   ah, cl           ; BIT 7 POSITION
```

```
2850 00000F33 C0E405            <1>         shl    ah, 5
2851 00000F36 A0[666E0000]      <1>         mov    al, [KB_FLAG_1]     ; GET SYSTEM SHIFT STATES BACK
2852 00000F3B 2473              <1>         and    al, 01110011b       ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
2853 00000F3D 08C4              <1>         or     ah, al               ; MERGE REMAINING BITS INTO AH
2854 00000F3F A0[686E0000]      <1>         mov    al, [KB_FLAG_3]     ; GET RIGHT CTL AND ALT
2855 00000F44 240C              <1>         and    al, 00001100b       ; ELIMINATE LC_E0 AND LC_E1
2856 00000F46 08C4              <1>         or     ah, al              ; OR THE SHIFT FLAGS TOGETHER
2857                            <1> _K3:
2858 00000F48 A0[656E0000]      <1>         mov    al, [KB_FLAG]       ; GET THE SHIFT STATUS FLAGS
2859                            <1>         ;jmp   short _KIO_EXIT     ; RETURN TO CALLER
2860 00000F4D EBD7              <1>         jmp    _KIO_EXIT
2861                            <1>
2862                            <1>         ;----- ASCII CHARACTER
2863                            <1> _K1E:
2864 00000F4F E8AE000000        <1>         call   _K1S                ; GET A CHARACTER FROM THE BUFFER (EXTENDED)
2865 00000F54 E821010000        <1>         call   _KIO_E_XLAT         ; ROUTINE TO XLATE FOR EXTENDED CALLS
2866 00000F59 EBCB              <1>         jmp    short _KIO_EXIT         ; GIVE IT TO THE CALLER
2867                            <1> _K1:
2868 00000F5B E8A2000000        <1>         call   _K1S                ; GET A CHARACTER FROM THE BUFFER
2869 00000F60 E820010000        <1>         call   _KIO_S_XLAT         ; ROUTINE TO XLATE FOR STANDARD CALLS
2870 00000F65 72F4              <1>         jc     short _K1           ; CARRY SET MEANS TROW CODE AWAY
2871                            <1> _K1A:
2872 00000F67 EBBD              <1>         jmp    short _KIO_EXIT     ; RETURN TO CALLER
2873                            <1>
2874                            <1>         ;----- ASCII STATUS
2875                            <1> _K2E:
2876 00000F69 E8DF000000        <1>         call   _K2S                ; TEST FOR CHARACTER IN BUFFER (EXTENDED)
2877 00000F6E 7420              <1>         jz     short _K2B          ; RETURN IF BUFFER EMPTY
2878 00000F70 9C                <1>         pushf                      ; SAVE ZF FROM TEST
2879 00000F71 E804010000        <1>         call   _KIO_E_XLAT         ; ROUTINE TO XLATE FOR EXTENDED CALLS
2880 00000F76 EB17              <1>         jmp    short _K2A           ; GIVE IT TO THE CALLER
2881                            <1> _K2:
2882 00000F78 E8D0000000        <1>         call    K2S                ; TEST FOR CHARACTER IN BUFFER
2883 00000F7D 7411              <1>         jz     short _K2B          ; RETURN IF BUFFER EMPTY
2884 00000F7F 9C                <1>         pushf                      ; SAVE ZF FROM TEST
2885 00000F80 E800010000        <1>         call   _KIO_S_XLAT         ; ROUTINE TO XLATE FOR STANDARD CALLS
2886 00000F85 7308              <1>         jnc    short _K2A          ; CARRY CLEAR MEANS PASS VALID CODE
2887 00000F87 9D                <1>         popf                       ; INVALID CODE FOR THIS TYPE OF CALL
2888 00000F88 E875000000        <1>         call   _K1S                ; THROW THE CHARACTER AWAY
2889 00000F8D EBE9              <1>         jmp    short _K2           ; GO LOOK FOR NEXT CHAR, IF ANY
2890                            <1> _K2A:
2891 00000F8F 9D                <1>         popf                       ; RESTORE ZF FROM TEST
2892                            <1> _K2B:
2893                            <1>         ; 02/01/2017
2894 00000F90 FA                <1>         cli
2895                            <1>         ;; mov byte [intflg], 0 ;; 15/01/2017
2896                            <1>         ;
2897                            <1>         ;pop   ecx                 ; RECOVER REGISTER
2898 00000F91 5B                <1>         pop    ebx                 ; RECOVER REGISTER
2899 00000F92 1F                <1>         pop    ds                  ; RECOVER SEGMENT
2900                            <1>         ; (*) 29/05/2016
2901                            <1>         ; (*) retf 4               ; THROW AWAY (e)FLAGS
2902 00000F93 7208              <1>         jc     short _k2d
2903 00000F95 7505              <1>         jnz    short _k2c
2904 00000F97 804C240840        <1>         or     byte [esp+8], 01000000b   ; set zero flag bit of eflags register
2905                            <1> _k2c:
2906 00000F9C CF                <1>         iretd
2907                            <1> _k2d:
2908                            <1>         ; 29/05/2016 -set carry flag on stack-
2909                            <1>         ; [esp] = EIP
2910                            <1>         ; [esp+4] = CS
2911                            <1>         ; [esp+8] = E-FLAGS
2912 00000F9D 804C240801        <1>         or     byte [esp+8], 1  ; set carry bit of eflags register
2913                            <1>         ; [esp+12] = ESP (user)
2914                            <1>         ; [esp+16] = SS (User)
2915 00000FA2 CF                <1>         iretd
2916                            <1>
2917                            <1>         ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
2918                            <1>         ; (OUTER-PRIVILEGE-LEVEL)
2919                            <1>         ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
2920                            <1>         ; // RETF instruction:
2921                            <1>         ;
2922                            <1>         ; IF OperandMode=32 THEN
2923                            <1>         ;    Load CS:EIP from stack;
2924                            <1>         ;    Set CS RPL to CPL;
2925                            <1>         ;    Increment eSP by 8 plus the immediate offset if it exists;
2926                            <1>         ;    Load SS:eSP from stack;
2927                            <1>         ; ELSE (* OperandMode=16 *)
2928                            <1>         ;    Load CS:IP from stack;
2929                            <1>         ;    Set CS RPL to CPL;
2930                            <1>         ;    Increment eSP by 4 plus the immediate offset if it exists;
2931                            <1>         ;    Load SS:eSP from stack;
2932                            <1>         ; FI;
2933                            <1>         ;
2934                            <1>         ; //
2935                            <1>
2936                            <1>         ;----- SET TYPAMATIC RATE AND DELAY
2937                            <1> _K300:
2938 00000FA3 3C05              <1>         cmp    al, 5               ; CORRECT FUNCTION CALL?
2939                            <1>         ;jne   short _KIO_EXIT     ; NO, RETURN
2940 00000FA5 0F857BFFFFFF      <1>         jne    _KIO_EXIT
2941 00000FAB F6C3E0            <1>         test   bl, 0E0h            ; TEST FOR OUT-OF-RANGE RATE
2942 00000FAE 0F8572FFFFFF      <1>         jnz    _KIO_EXIT           ; RETURN IF SO
2943 00000FB4 F6C7FC            <1>         test   BH, 0FCh            ; TEST FOR OUT-OF-RANGE DELAY
2944 00000FB7 0F8569FFFFFF      <1>         jnz    _KIO_EXIT           ; RETURN IF SO
2945 00000FBD B0F3              <1>         mov    al, KB_TYPA_RD      ; COMMAND FOR TYPAMATIC RATE/DELAY
2946 00000FBF E87E060000        <1>         call   SND_DATA            ; SEND TO KEYBOARD
2947                            <1>         ;mov   cx, 5               ; SHIFT COUNT
2948                            <1>         ;shl   bh, cl              ; SHIFT DELAY OVER
2949 00000FC4 C0E705            <1>         shl    bh, 5
2950 00000FC7 88D8              <1>         mov    al, bl              ; PUT IN RATE
2951 00000FC9 08F8              <1>         or     al, bh              ; AND DELAY
2952 00000FCB E872060000        <1>         call   SND_DATA            ; SEND TO KEYBOARD
2953 00000FD0 E951FFFFFF        <1>         jmp    _KIO_EXIT           ; RETURN TO CALLER
2954                            <1>
```

```
                        <1>       ;----- WRITE TO KEYBOARD BUFFER
                        <1> _K500:
00000FD5 56             <1>       push   esi                ; SAVE SI (esi)
00000FD6 FA             <1>       cli                       ;
00000FD7 8B1D[766E0000] <1>       mov    ebx, [BUFFER_TAIL] ; GET THE 'IN TO' POINTER TO THE BUFFER
00000FDD 89DE           <1>       mov    esi, ebx           ; SAVE A COPY IN CASE BUFFER NOT FULL
00000FDF E8D1000000     <1>       call   _K4                ; BUMP THE POINTER TO SEE IF BUFFER IS FULL
00000FE4 3B1D[726E0000] <1>       cmp    ebx, [BUFFER_HEAD] ; WILL THE BUFFER OVERRUN IF WE STORE THIS?
00000FEA 740D           <1>       je     short _K502        ; YES - INFORM CALLER OF ERROR
00000FEC 66890E         <1>       mov    [esi], cx          ; NO - PUT ASCII/SCAN CODE INTO BUFFER
00000FEF 891D[766E0000] <1>       mov    [BUFFER_TAIL], ebx ; ADJUST 'IN TO' POINTER TO REFLECT CHANGE
00000FF5 28C0           <1>       sub    al, al             ; TELL CALLER THAT OPERATION WAS SUCCESSFUL
00000FF7 EB02           <1>       jmp    short _K504        ; SUB INSTRUCTION ALSO RESETS CARRY FLAG
                        <1> _K502:
00000FF9 B001           <1>       mov    al, 01h            ; BUFFER FULL INDICATION
                        <1> _K504:
00000FFB FB             <1>       sti
00000FFC 5E             <1>       pop    esi                ; RECOVER SI (esi)
00000FFD 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:
00001002 FA             <1>       cli    ; 03/12/2014
00001003 8B1D[726E0000] <1>       mov    ebx, [BUFFER_HEAD] ; GET POINTER TO HEAD OF BUFFER
00001009 3B1D[766E0000] <1>       cmp    ebx, [BUFFER_TAIL] ; TEST END OF BUFFER
                        <1>       ;jne   short _K1U         ; IF ANYTHING IN BUFFER SKIP INTERRUPT
0000100F 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
00001011 FB             <1>       sti                       ; INTERRUPTS BACK ON DURING LOOP
00001012 90             <1>       nop                       ; ALLOW AN INTERRUPT TO OCCUR
                        <1> _K1U:
00001013 FA             <1>       cli                       ; INTERRUPTS BACK OFF
00001014 8B1D[726E0000] <1>       mov    ebx, [BUFFER_HEAD] ; GET POINTER TO HEAD OF BUFFER
0000101A 3B1D[766E0000] <1>       cmp   ebx, [BUFFER_TAIL]  ; TEST END OF BUFFER
                        <1> _k1x:
00001020 53             <1>       push   ebx                ; SAVE ADDRESS
00001021 9C             <1>       pushf                     ; SAVE FLAGS
00001022 E8CF060000     <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
00001027 8A1D[676E0000] <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
0000102D 30C3           <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
0000102F 80E307         <1>       and    bl, 07h ; KB_LEDS  ; ISOLATE INDICATOR BITS
00001032 7406           <1>       jz     short _K1V         ; IF NO CHANGE BYPASS UPDATE
00001034 E869060000     <1>       call   SND_LED1
00001039 FA             <1>       cli                       ; DISABLE INTERRUPTS
                        <1> _K1V:
0000103A 9D             <1>       popf                      ; RESTORE FLAGS
0000103B 5B             <1>       pop    ebx                ; RESTORE ADDRESS
0000103C 74D3           <1>       je    short _K1T          ; LOOP UNTIL SOMETHING IN BUFFER
                        <1>       ;
0000103E 668B03         <1>       mov    ax, [ebx]          ; GET SCAN CODE AND ASCII CODE
00001041 E86F000000     <1>       call  _K4                 ; MOVE POINTER TO NEXT POSITION
00001046 891D[726E0000] <1>       mov   [BUFFER_HEAD], ebx  ; STORE VALUE IN VARIABLE
0000104C C3             <1>       retn                      ; RETURN
                        <1>
                        <1>       ;----- READ THE KEY TO SEE IF ONE IS PRESENT -----
                        <1> _K2S:
0000104D FA             <1>       cli                       ; INTERRUPTS OFF
0000104E 8B1D[726E0000] <1>       mov   ebx, [BUFFER_HEAD]  ; GET HEAD POINTER
00001054 3B1D[766E0000] <1>       cmp   ebx, [BUFFER_TAIL]  ; IF EQUAL (Z=1) THEN NOTHING THERE
0000105A 668B03         <1>       mov    ax, [ebx]
0000105D 9C             <1>       pushf                     ; SAVE FLAGS
                        <1>       ;push  ax                 ; SAVE CODE
                        <1>       ; 12/04/2021
0000105E 50             <1>       push   eax
0000105F E892060000     <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
00001064 8A1D[676E0000] <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
0000106A 30C3           <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
0000106C 80E307         <1>       and    bl, 07h ; KB_LEDS  ; ISOLATE INDICATOR BITS
0000106F 7405           <1>       jz     short _K2T         ; IF NO CHANGE BYPASS UPDATE
00001071 E815060000     <1>       call   SND_LED            ; GO TURN ON MODE INDICATORS
                        <1> _K2T:
                        <1>       ;pop   ax                 ; RESTORE CODE
                        <1>       ; 12/04/2021
00001076 58             <1>       pop    eax
00001077 9D             <1>       popf                      ; RESTORE FLAGS
00001078 FB             <1>       sti                       ; INTERRUPTS BACK ON
00001079 C3             <1>       retn                      ; RETURN
                        <1>
                        <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR EXTENDED CALLS -----
                        <1> _KIO_E_XLAT:
0000107A 3CF0           <1>       cmp    al, 0F0h           ; IS IT ONE OF THE FILL-INs?
0000107C 7506           <1>       jne    short _KIO_E_RET   ; NO, PASS IT ON
0000107E 08E4           <1>       or     ah, ah             ; AH = 0 IS SPECIAL CASE
00001080 7402           <1>       jz     short _KIO_E_RET    ; PASS THIS ON UNCHANGED
00001082 30C0           <1>       xor    al, al             ; OTHERWISE SET AL = 0
                        <1> _KIO_E_RET:
00001084 C3             <1>       retn                      ; GO BACK
                        <1>
                        <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR STANDARD CALLS -----
                        <1> _KIO_S_XLAT:
00001085 80FCE0         <1>       cmp    ah, 0E0h           ; IS IT KEYPAD ENTER OR / ?
00001088 750F           <1>       jne    short _KIO_S2      ; NO, CONTINUE
0000108A 3C0D           <1>       cmp    al, 0Dh            ; KEYPAD ENTER CODE?
0000108C 7408           <1>       je     short _KIO_S1      ; YES, MASSAGE A BIT
0000108E 3C0A           <1>       cmp    al, 0Ah            ; CTRL KEYPAD ENTER CODE?
00001090 7404           <1>       je     short _KIO_S1      ; YES, MASSAGE THE SAME
00001092 B435           <1>       mov    ah, 35h            ; NO, MUST BE KEYPAD /
                        <1> _kio_ret: ; 03/12/2014
00001094 F8             <1>       clc
```

```
3060 00001095 C3                <1>       retn
3061                            <1>       ;jmp   short _KIO_USE      ; GIVE TO CALLER
3062                            <1> _KIO_S1:
3063 00001096 B41C              <1>       mov    ah, 1Ch             ; CONVERT TO COMPATIBLE OUTPUT
3064                            <1>       ;jmp   short _KIO_USE      ; GIVE TO CALLER
3065 00001098 C3                <1>       retn
3066                            <1> _KIO_S2:
3067 00001099 80FC84            <1>       cmp    ah, 84h             ; IS IT ONE OF EXTENDED ONES?
3068 0000109C 7715              <1>       ja     short _KIO_DIS      ; YES, THROW AWAY AND GET ANOTHER CHAR
3069 0000109E 3CF0              <1>       cmp    al, 0F0h            ; IS IT ONE OF THE FILL-INs?
3070 000010A0 7506              <1>       jne    short _KIO_S3       ; NO, TRY LAST TEST
3071 000010A2 08E4              <1>       or     ah, ah              ; AH = 0 IS SPECIAL CASE
3072 000010A4 740C              <1>       jz     short _KIO_USE      ; PASS THIS ON UNCHANGED
3073 000010A6 EB0B              <1>       jmp    short _KIO_DIS      ; THROW AWAY THE REST
3074                            <1> _KIO_S3:
3075 000010A8 3CE0              <1>       cmp    al, 0E0h            ; IS IT AN EXTENSION OF A PREVIOUS ONE?
3076                            <1>       ;jne   short _KIO_USE      ; NO, MUST BE A STANDARD CODE
3077 000010AA 75E8              <1>       jne    short _kio_ret
3078 000010AC 08E4              <1>       or     ah, ah              ; AH = 0 IS SPECIAL CASE
3079 000010AE 7402              <1>       jz     short _KIO_USE      ; JUMP IF AH = 0
3080 000010B0 30C0              <1>       xor    al, al              ; CONVERT TO COMPATIBLE OUTPUT
3081                            <1>       ;jmp   short _KIO_USE      ; PASS IT ON TO CALLER
3082                            <1> _KIO_USE:
3083                            <1>       ;clc                       ; CLEAR CARRY TO INDICATE GOOD CODE
3084 000010B2 C3                <1>       retn                       ; RETURN
3085                            <1> _KIO_DIS:
3086 000010B3 F9                <1>       stc                        ; SET CARRY TO INDICATE DISCARD CODE
3087 000010B4 C3                <1>       retn                       ; RETURN
3088                            <1>
3089                            <1>       ;----- INCREMENT BUFFER POINTER ROUTINE -----
3090                            <1> _K4:
3091 000010B5 43                <1>       inc    ebx
3092 000010B6 43                <1>       inc    ebx                 ; MOVE TO NEXT WORD IN LIST
3093 000010B7 3B1D[6E6E0000]    <1>       cmp    ebx, [BUFFER_END]   ; AT END OF BUFFER?
3094                            <1>       ;jne   short _K5           ; NO, CONTINUE
3095 000010BD 7206              <1>       jb     short _K5
3096 000010BF 8B1D[6A6E0000]    <1>       mov    ebx, [BUFFER_START] ; YES, RESET TO BUFFER BEGINNING
3097                            <1> _K5:
3098 000010C5 C3                <1>       retn
3099                            <1>
3100                            <1> ; 20/02/2015
3101                            <1> ; 05/12/2014
3102                            <1> ; 26/08/2014
3103                            <1> ; KEYBOARD (HARDWARE) INTERRUPT -  IRQ LEVEL 1
3104                            <1> ; (INT_09h - Retro UNIX 8086 v1 - U9.ASM, 07/03/2014)
3105                            <1> ;
3106                            <1> ; Derived from "KB_INT_1" procedure of IBM "pc-at"
3107                            <1> ; rombios source code (06/10/1985)
3108                            <1> ; 'keybd.asm', HARDWARE INT 09h - (IRQ Level 1)
3109                            <1>
3110                            <1> ; EQUATES (IBM PC-XT-286 BIOS, 1986, 'POSQEQU.INC')
3111                            <1>
3112                            <1> ;--------- 8042 COMMANDS --------------------------------------------------------
3113                            <1> ENA_KBD      equ    0AEh         ; ENABLE KEYBOARD COMMAND
3114                            <1> DIS_KBD      equ    0ADh         ; DISABLE KEYBOARD COMMAND
3115                            <1> SHUT_CMD     equ    0FEh         ; CAUSE A SHUTDOWN COMMAND
3116                            <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
3117                            <1> STATUS_PORT  equ    064h         ; 8042 STATUS PORT
3118                            <1> INPT_BUF_FULL equ  00000010b     ; 1 = +INPUT BUFFER FULL
3119                            <1> PORT_A       equ    060h         ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
3120                            <1> ;---------- 8042 KEYBOARD RESPONSE ----------------------------------------------
3121                            <1> KB_ACK       equ    0FAh         ; ACKNOWLEDGE PROM TRANSMISSION
3122                            <1> KB_RESEND    equ    0FEh         ; RESEND REQUEST
3123                            <1> KB_OVER_RUN  equ    0FFh         ; OVER RUN SCAN CODE
3124                            <1> ;---------- KEYBOARD/LED COMMANDS -----------------------------------------------
3125                            <1> KB_ENABLE    equ    0F4h         ; KEYBOARD ENABLE
3126                            <1> LED_CMD      equ    0EDh         ; LED WRITE COMMAND
3127                            <1> KB_TYPA_RD   equ    0F3h         ; TYPAMATIC RATE/DELAY COMMAND
3128                            <1> ;---------- KEYBOARD SCAN CODES -------------------------------------------------
3129                            <1> NUM_KEY      equ    69           ; SCAN CODE FOR      NUMBER LOCK KEY
3130                            <1> SCROLL_KEY   equ    70           ; SCAN CODE FOR      SCROLL LOCK KEY
3131                            <1> ALT_KEY      equ    56           ; SCAN CODE FOR      ALTERNATE SHIFT KEY
3132                            <1> CTL_KEY      equ    29           ; SCAN CODE FOR      CONTROL KEY
3133                            <1> CAPS_KEY     equ    58           ; SCAN CODE FOR      SHIFT LOCK KEY
3134                            <1> DEL_KEY      equ    83           ; SCAN CODE FOR      DELETE KEY
3135                            <1> INS_KEY      equ    82           ; SCAN CODE FOR      INSERT KEY
3136                            <1> LEFT_KEY     equ    42           ; SCAN CODE FOR      LEFT SHIFT
3137                            <1> RIGHT_KEY    equ    54           ; SCAN CODE FOR      RIGHT SHIFT
3138                            <1> SYS_KEY      equ    84           ; SCAN CODE FOR      SYSTEM KEY
3139                            <1> ;---------- ENHANCED KEYBOARD SCAN CODES ----------------------------------------
3140                            <1> ID_1         equ    0ABh         ; 1ST ID CHARACTER FOR KBX
3141                            <1> ID_2         equ    041h         ; 2ND ID CHARACTER FOR KBX
3142                            <1> ID_2A        equ    054h         ; ALTERNATE 2ND ID CHARACTER FOR KBX
3143                            <1> F11_M        equ    87           ; F11 KEY MAKE
3144                            <1> F12_M        equ    88           ; F12 KEY MAKE
3145                            <1> MC_E0        equ    224          ; GENERAL MARKER CODE
3146                            <1> MC_E1        equ    225          ; PAUSE KEY MARKER CODE
3147                            <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG-----------------------------------------
3148                            <1> RIGHT_SHIFT  equ    00000001b    ; RIGHT SHIFT KEY DEPRESSED
3149                            <1> LEFT_SHIFT   equ    00000010b    ; LEFT SHIFT KEY DEPRESSED
3150                            <1> CTL_SHIFT    equ    00000100b    ; CONTROL SHIFT KEY DEPRESSED
3151                            <1> ALT_SHIFT    equ    00001000b    ; ALTERNATE SHIFT KEY DEPRESSED
3152                            <1> SCROLL_STATE equ    00010000b    ; SCROLL LOCK STATE IS ACTIVE
3153                            <1> NUM_STATE    equ    00100000b    ; NUM LOCK STATE IS ACTIVE
3154                            <1> CAPS_STATE   equ    01000000b    ; CAPS LOCK STATE IS ACTIVE
3155                            <1> INS_STATE    equ    10000000b    ; INSERT STATE IS ACTIVE
3156                            <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG_1 -------------------------------------
3157                            <1> L_CTL_SHIFT  equ    00000001b    ; LEFT CTL KEY DOWN
3158                            <1> L_ALT_SHIFT  equ    00000010b    ; LEFT ALT KEY DOWN
3159                            <1> SYS_SHIFT    equ    00000100b    ; SYSTEM KEY DEPRESSED AND HELD
3160                            <1> HOLD_STATE   equ    00001000b    ; SUSPEND KEY HAS BEEN TOGGLED
3161                            <1> SCROLL_SHIFT equ    00010000b    ; SCROLL LOCK KEY IS DEPRESSED
3162                            <1> NUM_SHIFT    equ    00100000b    ; NUM LOCK KEY IS DEPRESSED
3163                            <1> CAPS_SHIFT   equ    01000000b    ; CAPS LOCK KEY IS DEPRE55ED
3164                            <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:
000010C6 FB             <1>       sti                       ; ENABLE INTERRUPTS
                        <1>       ;push  ebp
000010C7 50             <1>       push   eax
000010C8 53             <1>       push   ebx
000010C9 51             <1>       push   ecx
000010CA 52             <1>       push   edx
000010CB 56             <1>       push   esi
000010CC 57             <1>       push   edi
000010CD 1E             <1>       push   ds
000010CE 06             <1>       push   es
000010CF FC             <1>       cld                       ; FORWARD DIRECTION
000010D0 66B81000       <1>       mov    ax, KDATA
000010D4 8ED8           <1>       mov    ds, ax
000010D6 8EC0           <1>       mov    es, ax
                        <1>       ;
                        <1>       ;-----WAIT FOR KEYBOARD DISABLE COMMAND TO BE ACCEPTED
000010D8 B0AD           <1>       mov    al, DIS_KBD        ; DISABLE THE KEYBOARD COMMAND
000010DA E851050000     <1>       call   SHIP_IT            ; EXECUTE DISABLE
000010DF FA             <1>       cli                       ; DISABLE INTERRUPTS
000010E0 B900000100     <1>       mov    ecx, 10000h        ; SET MAXIMUM TIMEOUT
                        <1> KB_INT_01:
000010E5 E464           <1>       in     al, STATUS_PORT    ; READ ADAPTER STATUS
000010E7 A802           <1>       test   al, INPT_BUF_FULL  ; CHECK INPUT BUFFER FULL STATUS BIT
000010E9 E0FA           <1>       loopnz KB_INT_01          ; WAIT FOR COMMAND TO BE ACCEPTED
                        <1>       ;
                        <1>       ;-----READ CHARACTER FROM KEYBOARD INTERFACE
000010EB 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>       ;                         ; EXIT HANDLES HARDWARE EOI AND ENABLE
                        <1>       ;
                        <1>       ;-----CHECK FOR A RESEND COMMAND TO KEYBOARD
                        <1> KB_INT_02:                      ; (AL)= SCAN CODE
000010ED FB             <1>       sti                       ; ENABLE INTERRUPTS AGAIN
000010EE 3CFE           <1>       cmp    al, KB_RESEND      ; IS THE INPUT A RESEND
000010F0 740E           <1>         je     short KB_INT_4   ; GO IF RESEND
                        <1>       ;
                        <1>       ;-----CHECK FOR RESPONSE TO A COMMAND TO KEYBOARD
000010F2 3CFA           <1>       cmp    al, KB_ACK         ; IS THE INPUT AN ACKNOWLEDGE
000010F4 7514           <1>         jne    short KB_INT_2   ; GO IF NOT
                        <1>       ;
                        <1>       ;-----A COMMAND TO THE KEYBOARD WAS ISSUED
000010F6 FA             <1>       cli                       ; DISABLE INTERRUPTS
000010F7 800D[676E0000]10 <1>     or     byte [KB_FLAG_2], KB_FA ; INDICATE ACK RECEIVED
                        <1>       ;jmp   K26                ; RETURN IF NOT ACK RETURNED FOR DATA)
                        <1>       ; 12/04/2021
000010FE EB76           <1>       jmp    short ID_EX  ; K26
                        <1>       ;
                        <1>       ;-----RESEND THE LAST BYTE
                        <1> KB_INT_4:
00001100 FA             <1>       cli                       ; DISABLE INTERRUPTS
00001101 800D[676E0000]20 <1>     or     byte [KB_FLAG_2], KB_FE ; INDICATE RESEND RECEIVED
                        <1>       ;jmp   K26                ; RETURN IF NOT ACK RETURNED FOR DATA)
                        <1>       ; 12/04/2021
00001108 EB6C           <1>       jmp    short ID_EX  ; K26
                        <1>       ;
```

```
                        <1> ;-----      UPDATE MODE INDICATORS IF CHANGE IN STATE
                        <1> KB_INT_2:
                        <1>       ;push  ax                 ; SAVE DATA IN
                        <1>       ; 12/04/2021
0000110A 50             <1>       push   eax
0000110B E8E6050000     <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
00001110 8A1D[676E0000] <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
00001116 30C3           <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
00001118 80E307         <1>       and    bl, KB_LEDS        ; ISOLATE INDICATOR BITS
0000111B 7405           <1>       jz     short UP0          ; IF NO CHANGE BYPASS UPDATE
0000111D E869050000     <1>       call   SND_LED                  ; GO TURN ON MODE INDICATORS
                        <1> UP0:
                        <1>       ;pop   ax                 ; RESTORE DATA IN
                        <1>       ; 12/04/2021
00001122 58             <1>       pop    eax
                        <1> ;--------------------------------------------------------------------------
                        <1> ;     START OF KEY PROCESSING                                        ;
                        <1> ;--------------------------------------------------------------------------
00001123 88C4           <1>       mov    ah, al             ; SAVE SCAN CODE IN AH ALSO
                        <1>       ;
                        <1>       ;----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
00001125 3CFF           <1>       cmp    al, KB_OVER_RUN           ; IS THIS AN OVERRUN CHAR
                        <1>         ;je  K62                ; BUFFER_FULL_BEEP
                        <1>       ; 12/04/2021
00001127 7505           <1>       jne    short K16
00001129 E9EE040000     <1>       jmp    K62
                        <1> K16:
0000112E 8A3D[686E0000] <1>       mov    bh, [KB_FLAG_3]           ; LOAD FLAGS FOR TESTING
                        <1>       ;
                        <1>       ;----- TEST TO SEE IF A READ_ID IS IN PROGRESS
00001134 F6C7C0         <1>       test   bh, RD_ID+LC_AB    ; ARE WE DOING A READ ID?
00001137 7442           <1>       jz     short NOT_ID       ; CONTINUE IF NOT
00001139 7914           <1>       jns    short TST_ID_2            ; IS THE RD_ID FLAG ON?
0000113B 3CAB           <1>       cmp    al, ID_1           ; IS THIS THE 1ST ID CHARACTER?
0000113D 7507           <1>       jne    short RST_RD_ID
0000113F 800D[686E0000]40 <1>     or     byte [KB_FLAG_3], LC_AB ; INDICATE 1ST ID WAS OK
                        <1> RST_RD_ID:
00001146 8025[686E0000]7F <1>     and    byte [KB_FLAG_3], ~RD_ID ; RESET THE READ ID FLAG
0000114D EB27           <1>       jmp    short ID_EX        ; AND EXIT
                        <1>       ; 12/04/2021
                        <1>       ;jmp   K26
                        <1>       ;
                        <1> TST_ID_2:
0000114F 8025[686E0000]BF <1>     and    byte [KB_FLAG_3], ~LC_AB ; RESET FLAG
00001156 3C54           <1>       cmp    al, ID_2A          ; IS THIS THE 2ND ID CHARACTER?
00001158 7415           <1>       je     short KX_BIT       ; JUMP IF SO
0000115A 3C41           <1>       cmp    al, ID_2           ; IS THIS THE 2ND ID CHARACTER?
0000115C 7518           <1>       jne    short ID_EX        ; LEAVE IF NOT
                        <1>       ; 12/04/2021
                        <1>       ;jne   K26
                        <1>       ;
                        <1>       ;----- A READ ID SAID THAT IT WAS ENHANCED KEYBOARD
0000115E F6C720         <1>       test   bh, SET_NUM_LK     ; SHOULD WE SET NUM LOCK?
00001161 740C           <1>       jz    short KX_BIT        ; EXIT IF NOT
00001163 800D[656E0000]20 <1>     or     byte [KB_FLAG], NUM_STATE ; FORCE NUM LOCK ON
0000116A E81C050000     <1>       call   SND_LED            ; GO SET THE NUM LOCK INDICATOR
                        <1> KX_BIT:
0000116F 800D[686E0000]10 <1>     or     byte [KB_FLAG_3], KBX ; INDICATE ENHANCED KEYBOARD WAS FOUND
00001176 E9CF010000     <1> ID_EX: jmp     K26             ; EXIT
                        <1>       ;
                        <1> NOT_ID:
0000117B 3CE0           <1>       cmp    al, MC_E0          ; IS THIS THE GENERAL MARKER CODE?
0000117D 7509           <1>       jne    short TEST_E1
0000117F 800D[686E0000]12 <1>     or     byte [KB_FLAG_3], LC_E0+KBX ; SET FLAG BIT, SET KBX, AND
00001186 EB0B           <1>       jmp    short EXIT         ; THROW AWAY THIS CODE
                        <1>       ; 12/04/2021
                        <1>       ;jmp   K26A
                        <1> TEST_E1:
00001188 3CE1           <1>       cmp    al, MC_E1          ; IS THIS THE PAUSE KEY?
0000118A 750C           <1>       jne    short NOT_HC
0000118C 800D[686E0000]11 <1>     or     byte [KB_FLAG_3], LC_E1+KBX ; SET FLAG BIT, SET KBX, AND
00001193 E9B9010000     <1> EXIT: jmp    K26A               ; THROW AWAY THIS CODE
                        <1>       ;
                        <1> NOT_HC:
00001198 247F           <1>       and    al, 07Fh           ; TURN OFF THE BREAK BIT
0000119A F6C702         <1>       test   bh, LC_E0          ; LAST CODE THE E0 MARKER CODE
0000119D 7410           <1>       jz     short NOT_LC_E0          ; JUMP IF NOT
                        <1>       ;
0000119F BF[526D0000]   <1>       mov    edi, _K6+6         ; IS THIS A SHIFT KEY?
000011A4 AE             <1>       scasb
                        <1>       ;je    K26 ; K16B             ; YES, THROW AWAY & RESET FLAG
                        <1>       ; 12/04/2021
000011A5 745F           <1>       je     short K16B ; K26
000011A7 AE             <1>       scasb
000011A8 7571           <1>       jne    short K16A         ; NO, CONTINUE KEY PROCESSING
                        <1>       ;jmp   short K16B         ; YES, THROW AWAY & RESET FLAG
000011AA E99B010000     <1>       jmp    K26
                        <1>       ;
                        <1> NOT_LC_E0:
000011AF F6C701         <1>       test   bh, LC_E1          ; LAST CODE THE E1 MARKER CODE?
000011B2 7429           <1>       jz     short T_SYS_KEY          ; JUMP IF NOT
000011B4 B904000000     <1>       mov    ecx, 4             ; LENGHT OF SEARCH
000011B9 BF[506D0000]   <1>       mov    edi, _K6+4         ; IS THIS AN ALT, CTL, OR SHIFT?
000011BE F2AE           <1>       repne  scasb              ; CHECK IT
000011C0 74D1           <1>       je     short EXIT         ; THROW AWAY IF SO
                        <1>       ; 12/04/2021
                        <1>       ;je    K26A
                        <1>       ;
000011C2 3C45           <1>       cmp    al, NUM_KEY        ; IS IT THE PAUSE KEY?
000011C4 7540           <1>       jne    short K16B         ; NO, THROW AWAY & RESET FLAG
                        <1>       ; 12/04/2021
                        <1>       ;jne   K26
000011C6 F6C480         <1>       test   ah, 80h                ; YES, IS IT THE BREAK OF THE KEY?
                        <1>       ;jnz   short K16B         ; YES, THROW THIS AWAY, TOO
000011C9 0F857B010000   <1>       jnz    K26
```

```
                                <1>         ; 20/02/2015
000011CF F605[666E0000]08       <1>       test  byte [KB_FLAG_1],HOLD_STATE ; NO, ARE WE PAUSED ALREADY?
000011D6 752E                   <1>       jnz   short K16B          ; YES, THROW AWAY
                                <1>       ; 12/04/2021
                                <1>       ;jnz  K26
000011D8 E9B1020000             <1>       jmp    K39P             ; NO, THIS IS THE REAL PAUSE STATE
                                <1>       ;
                                <1>       ;----- TEST FOR SYSTEM KEY
                                <1> T_SYS_KEY:
000011DD 3C54                   <1>       cmp   al, SYS_KEY         ; IS IT THE SYSTEM KEY?
000011DF 753A                   <1>       jnz   short K16A          ; CONTINUE IF NOT
                                <1>       ;
000011E1 F6C480                 <1>       test  ah, 80h                   ; CHECK IF THIS A BREAK CODE
000011E4 7525                   <1>       jnz   short K16C          ; DO NOT TOUCH SYSTEM INDICATOR IF TRUE
                                <1>       ;
000011E6 F605[666E0000]04       <1>       test  byte [KB_FLAG_1], SYS_SHIFT ; SEE IF IN SYSTEM KEY HELD DOWN
000011ED 7517                   <1>       jnz   short K16B          ; IF YES, DO NOT PROCESS SYSTEM INDICATOR
                                <1>       ; 12/04/2021
                                <1>       ;jnz  K26
                                <1>       ;
000011EF 800D[666E0000]04       <1>       or    byte [KB_FLAG_1], SYS_SHIFT ; INDICATE SYSTEM KEY DEPRESSED
000011F6 B020                   <1>       mov   al, EOI             ; END OF INTERRUPT COMMAND
000011F8 E620                   <1>       out   20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
                                <1>                                 ; INTERRUPT-RETURN-NO-EOI
000011FA B0AE                   <1>       mov   al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
000011FC E82F040000             <1>       call  SHIP_IT             ; EXECUTE ENABLE
                                <1>       ; !!! SYSREQ !!! function/system call (INTERRUPT) must be here !!!
                                <1>       ;MOV  AL, 8500H           ; FUNCTION VALUE FOR MAKE OF SYSTEM KEY
                                <1>       ;STI                      ; MAKE SURE INTERRUPTS ENABLED
                                <1>       ;INT  15H                 ; USER INTERRUPT
00001201 E957010000             <1>       jmp  K27A                 ; END PROCESSING
                                <1>       ;
00001206 E93F010000             <1> K16B: jmp   K26                 ; IGNORE SYSTEM KEY
                                <1>       ;
                                <1> K16C:
0000120B 8025[666E0000]FB       <1>       and   byte [KB_FLAG_1], ~SYS_SHIFT ; TURN OFF SHIFT KEY HELD DOWN
00001212 B020                   <1>       mov   al, EOI             ; END OF INTERRUPT COMMAND
00001214 E620                   <1>       out   20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
                                <1>                                 ; INTERRUPT-RETURN-NO-EOI
                                <1>       ;MOV  AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
                                <1>       ;CALL SHIP_IT             ; EXECUTE ENABLE
                                <1>       ;
                                <1>       ;MOV  AX, 8501H           ; FUNCTION VALUE FOR BREAK OF SYSTEM KEY
                                <1>       ;STI                      ; MAKE SURE INTERRUPTS ENABLED
                                <1>       ;INT  15H                 ; USER INTERRUPT
                                <1>       ;JMP  K27A                ; INGONRE SYSTEM KEY
                                <1>       ;
00001216 E93B010000             <1>       jmp    K27                ; IGNORE SYSTEM KEY
                                <1>       ;
                                <1>       ;----- TEST FOR SHIFT KEYS
                                <1> K16A:
0000121B 8A1D[656E0000]         <1>       mov   bl, [KB_FLAG]       ; PUT STATE FLAGS IN BL
00001221 BF[4C6D0000]           <1>       mov   edi, _K6            ; SHIFT KEY TABLE offset
00001226 B908000000             <1>       mov   ecx, _K6L           ; LENGTH
0000122B F2AE                   <1>       repne scasb               ; LOOK THROUGH THE TABLE FOR A MATCH
0000122D 88E0                   <1>       mov   al, ah              ; RECOVER SCAN CODE
                                <1>         ;jne K25                    ; IF NO MATCH, THEN SHIFT NOT FOUND
                                <1>       ; 12/04/2021
0000122F 7405                   <1>       je    short K17
00001231 E9FC000000             <1>       jmp   K25
                                <1>       ;
                                <1>       ;------      SHIFT KEY FOUND
                                <1> K17:
00001236 81EF[4D6D0000]         <1>       sub   edi, _K6+1          ; ADJUST PTR TO SCAN CODE MATCH
0000123C 8AA7[546D0000]         <1>       mov   ah, [edi+_K7]       ; GET MASK INTO AH
00001242 B102                   <1>       mov   cl, 2               ; SETUP COUNT FOR FLAG SHIFTS
00001244 A880                   <1>       test  al, 80h             ; TEST FOR BREAK KEY
                                <1>       ;jnz  short K23           ; JUMP OF BREAK
                                <1>       ; 12/04/2021
00001246 7405                   <1>       jz    short K17C
00001248 E981000000             <1>       jmp   K23
                                <1>       ;
                                <1>       ;----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
                                <1> K17C:
0000124D 80FC10                 <1>       cmp   ah, SCROLL_SHIFT
00001250 7324                   <1>       jae   short K18           ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
                                <1>       ;
                                <1>       ;----- PLAIN SHIFT KEY, SET SHIFT ON
00001252 0825[656E0000]         <1>       or    [KB_FLAG], ah       ; TURN ON SHIFT BIT
00001258 A80C                   <1>         test al, CTL_SHIFT+ALT_SHIFT ; IS IT ALT OR CTRL?
                                <1>       ;;jnz  short K17D         ; YES, MORE FLAGS TO SET
                                <1>       ;jz   K26                 ; NO, INTERRUPT RETURN
                                <1>       ; 12/04/2021
0000125A 7415                   <1>       jz    short k17f
                                <1> K17D:
0000125C F6C702                 <1>       test  bh, LC_E0           ; IS THIS ONE OF NEW KEYS?
0000125F 7408                   <1>       jz    short K17E          ; NO, JUMP
00001261 0825[686E0000]         <1>       or    [KB_FLAG_3], ah     ; SET BITS FOR RIGHT CTRL, ALT
                                <1>       ;jmp  K26                 ; INTERRUPT RETURN
                                <1>       ; 12/04/2021
00001267 EB08                   <1>       jmp   short k17f
                                <1> K17E:
00001269 D2EC                   <1>       shr   ah, cl              ; MOVE FLAG BITS TWO POSITIONS
0000126B 0825[666E0000]         <1>       or    [KB_FLAG_1], ah     ; SET BITS FOR LEFT CTRL, ALT
                                <1> k17f: ; 12/04/2021
00001271 E9D4000000             <1>       jmp   K26
                                <1>       ;
                                <1>       ;----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
                                <1> K18:                            ; SHIFT-TOGGLE
00001276 F6C304                 <1>       test  bl, CTL_SHIFT       ; CHECK CTL SHIFT STATE
00001279 7402                   <1>       jz    short K18A          ; JUMP IF NOT CTL STATE
                                <1>       ;jnz  K25                 ; JUMP IF CTL STATE
                                <1>       ; 12/04/2021
0000127B EB1C                   <1>       jmp   short k20a ; K25
                                <1> K18A:
```

```
3480 0000127D 3C52              <1>       cmp   al, INS_KEY         ; CHECK FOR INSERT KEY
3481 0000127F 7522              <1>       jne   short K22           ; JUMP IF NOT INSERT KEY
3482 00001281 F6C308            <1>       test  bl, ALT_SHIFT       ; CHECK FOR ALTERNATE SHIFT
3483 00001284 7402              <1>       jz    short K18B          ; JUMP IF NOT ALTERNATE SHIFT
3484                            <1>       ;jnz  K25                 ; JUMP IF ALTERNATE SHIFT
3485                            <1>       ; 12/04/2021
3486 00001286 EB11              <1>       jmp   short k20a ; K25
3487                            <1> K18B:
3488 00001288 F6C702            <1>       test  bh, LC_E0 ;20/02/2015 ; IS THIS NEW INSERT KEY?
3489 0000128B 7516              <1>       jnz   short K22           ; YES, THIS ONE'S NEVER A '0'
3490                            <1> K19:
3491 0000128D F6C320            <1>       test  bl, NUM_STATE             ; CHECK FOR BASE STATE
3492 00001290 750C              <1>       jnz   short K21           ; JUMP IF NUM LOCK IS ON
3493 00001292 F6C303            <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE
3494 00001295 740C              <1>       jz    short K22           ; JUMP IF BASE STATE
3495                            <1> K20:                            ; NUMERIC ZERO, NOT INSERT KEY
3496 00001297 88C4              <1>       mov   ah, al              ; PUT SCAN CODE BACK IN AH
3497                            <1> k20a: ; 12/04/2021
3498 00001299 E994000000        <1>       jmp    K25                ; NUMERAL '0', STNDRD. PROCESSING
3499                            <1> K21:                            ; MIGHT BE NUMERIC
3500 0000129E F6C303            <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT
3501 000012A1 74F4              <1>       jz    short K20           ; IS NUMERIC, STD. PROC.
3502                            <1>       ;
3503                            <1> K22:                            ; SHIFT TOGGLE KEY HIT; PROCESS IT
3504 000012A3 8425[666E0000]    <1>       test  ah, [KB_FLAG_1]     ; IS KEY ALREADY DEPRESSED
3505                            <1>       ;jnz  short K26           ; JUMP IF KEY ALREADY DEPRESSED
3506                            <1>       ; 12/04/2021
3507 000012A9 75C6              <1>       jnz   short k17f ; K26
3508                            <1> K22A:
3509 000012AB 0825[666E0000]    <1>       or    [KB_FLAG_1], ah     ; INDICATE THAT THE KEY IS DEPRESSED
3510 000012B1 3025[656E0000]    <1>       xor   [KB_FLAG], ah       ; TOGGLE THE SHIFT STATE
3511                            <1>       ;
3512                            <1>       ;----- TOGGLE LED IF CAPS, NUM  OR SCROLL KEY DEPRESSED
3513 000012B7 F6C470            <1>       test  ah, CAPS_SHIFT+NUM_SHIFT+SCROLL_SHIFT ; SHIFT TOGGLE?
3514 000012BA 7407              <1>       jz    short K22B          ; GO IF NOT
3515                            <1>       ;
3516                            <1>       ; 12/04/2021 (32 bit push/pop)
3517 000012BC 50                <1>       push  eax ; push ax       ; SAVE SCAN CODE AND SHIFT MASK
3518 000012BD E8C9030000        <1>       call  SND_LED                   ; GO TURN MODE INDICATORS ON
3519 000012C2 58                <1>       pop   eax ; pop ax        ; RESTORE SCAN CODE
3520                            <1> K22B:
3521 000012C3 3C52              <1>       cmp   al, INS_KEY         ; TEST FOR 1ST MAKE OF INSERT KEY
3522                            <1>       ;jne  short K26           ; JUMP IF NOT INSERT KEY
3523                            <1>       ; 12/04/2021
3524 000012C5 75AA              <1>       jne   short k17f ; K26
3525 000012C7 88C4              <1>       mov   ah, al              ; SCAN CODE IN BOTH HALVES OF AX
3526 000012C9 E999000000        <1>       jmp   K28                 ; FLAGS UPDATED, PROC. FOR BUFFER
3527                            <1>       ;
3528                            <1>       ;----- BREAK SHIFT FOUND
3529                            <1> K23:                            ; BREAK-SHIFT-FOUND
3530 000012CE 80FC10            <1>       cmp   ah, SCROLL_SHIFT    ; IS THIS A TOGGLE KEY
3531 000012D1 F6D4              <1>       not   ah                  ; INVERT MASK
3532 000012D3 7355              <1>       jae   short K24           ; YES, HANDLE BREAK TOGGLE
3533 000012D5 2025[656E0000]    <1>       and   [KB_FLAG], ah       ; TURN OFF SHIFT BIT
3534 000012DB 80FCFB            <1>       cmp   ah, ~CTL_SHIFT            ; IS THIS ALT OR CTL?
3535 000012DE 7730              <1>       ja    short K23D          ; NO, ALL DONE
3536                            <1>       ;
3537 000012E0 F6C702            <1>       test  bh, LC_E0           ; 2ND ALT OR CTL?
3538 000012E3 7408              <1>       jz    short K23A          ; NO, HANSLE NORMALLY
3539 000012E5 2025[686E0000]    <1>       and   [KB_FLAG_3], ah           ; RESET BIT FOR RIGHT ALT OR CTL
3540 000012EB EB08              <1>       jmp   short K23B          ; CONTINUE
3541                            <1> K23A:
3542 000012ED D2FC              <1>       sar   ah, cl              ; MOVE THE MASK BIT TWO POSITIONS
3543 000012EF 2025[666E0000]    <1>       and   [KB_FLAG_1], ah           ; RESET BIT FOR LEFT ALT AND CTL
3544                            <1> K23B:
3545 000012F5 88C4              <1>       mov   ah, al              ; SAVE SCAN CODE
3546 000012F7 A0[686E0000]      <1>       mov   al, [KB_FLAG_3]           ; GET RIGHT ALT & CTRL FLAGS
3547 000012FC D2E8              <1>       shr   al, cl              ; MOVE TO BITS 1 & 0
3548 000012FE 0A05[666E0000]    <1>       or    al, [KB_FLAG_1]           ; PUT IN LEFT ALPT & CTL FLAGS
3549 00001304 D2E0              <1>       shl   al, cl              ; MOVE BACK TO BITS 3 & 2
3550 00001306 240C              <1>       and   al, ALT_SHIFT+CTL_SHIFT ; FILTER OUT OTHER GARBAGE
3551 00001308 0805[656E0000]    <1>       or    [KB_FLAG], al       ; PUT RESULT IN THE REAL FLAGS
3552 0000130E 88E0              <1>       mov   al, ah
3553                            <1> K23D:
3554 00001310 3CB8              <1>       cmp   al, ALT_KEY+80h     ; IS THIS ALTERNATE SHIFT RELEASE
3555 00001312 7536              <1>       jne   short K26           ; INTERRUPT RETURN
3556                            <1>       ;
3557                            <1>       ;----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
3558 00001314 A0[696E0000]      <1>       mov   al, [ALT_INPUT]
3559 00001319 B400              <1>       mov   ah, 0               ; SCAN CODE OF 0
3560 0000131B 8825[696E0000]    <1>       mov   [ALT_INPUT], ah     ; ZERO OUT THE FIELD
3561 00001321 3C00              <1>       cmp   al, 0               ; WAS THE INPUT = 0?
3562 00001323 7425              <1>       je    short K26           ; INTERRUPT_RETURN
3563                            <1>       ; 29/01/2016
3564                            <1>       ;jmp  K61                 ; IT WASN'T, SO PUT IN BUFFER
3565 00001325 E9AB020000        <1>       jmp   _K60
3566                            <1>       ;
3567                            <1> K24:                            ; BREAK-TOGGLE
3568 0000132A 2025[666E0000]    <1>       and   [KB_FLAG_1], ah     ; INDICATE NO LONGER DEPRESSED
3569 00001330 EB18              <1>       jmp   short K26           ; INTERRUPT_RETURN
3570                            <1>       ;
3571                            <1>       ;----- TEST FOR HOLD STATE
3572                            <1>                                 ; AL, AH = SCAN CODE
3573                            <1> K25:                            ; NO-SHIFT-FOUND
3574 00001332 3C80              <1>       cmp   al, 80h             ; TEST FOR BREAK KEY
3575 00001334 7314              <1>       jae   short K26           ; NOTHING FOR BREAK CHARS FROM HERE ON
3576 00001336 F605[666E0000]08  <1>       test  byte [KB_FLAG_1], HOLD_STATE ; ARE WE IN HOLD STATE
3577 0000133D 7428              <1>       jz    short K28           ; BRANCH AROUND TEST IF NOT
3578 0000133F 3C45              <1>       cmp   al, NUM_KEY
3579 00001341 7407              <1>       je    short K26           ; CAN'T END HOLD ON NUM_LOCK
3580 00001343 8025[666E0000]F7  <1>       and   byte [KB_FLAG_1], ~HOLD_STATE ; TURN OFF THE HOLD STATE BIT
3581                            <1> K26:
3582 0000134A 8025[686E0000]FC  <1>       and   byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
3583                            <1> K26A:                           ; INTERRUPT-RETURN
3584 00001351 FA                <1>       cli                       ; TURN OFF INTERRUPTS
```

```
3585 00001352 B020                <1>       mov    al, EOI             ; END OF INTERRUPT COMMAND
3586 00001354 E620                <1>       out    20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
3587                              <1> K27:                               ; INTERRUPT-RETURN-NO-EOI
3588 00001356 B0AE                <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3589 00001358 E8D3020000          <1>       call   SHIP_IT             ; EXECUTE ENABLE
3590                              <1> K27A:
3591 0000135D FA                  <1>       cli                        ; DISABLE INTERRUPTS
3592                              <1>       ;;mov  byte [intflg], 0 ; 07/01/2017 ;; 15/01/2017
3593 0000135E 07                  <1>       pop    es                  ; RESTORE REGISTERS
3594 0000135F 1F                  <1>       pop    ds
3595 00001360 5F                  <1>       pop    edi
3596 00001361 5E                  <1>       pop    esi
3597 00001362 5A                  <1>       pop    edx
3598 00001363 59                  <1>       pop    ecx
3599 00001364 5B                  <1>       pop    ebx
3600 00001365 58                  <1>       pop    eax
3601                              <1>       ;pop   ebp
3602 00001366 CF                  <1>       iretd                      ; RETURN
3603                              <1>
3604                              <1>       ;----- NOT IN HOLD STATE
3605                              <1> K28:                               ; NO-HOLD-STATE
3606 00001367 3C58                <1>       cmp    al, 88              ; TEST FOR OUT-OF-RANGE SCAN CODES
3607 00001369 77DF                <1>       ja     short K26           ; IGNORE IF OUT-OF-RANGE
3608                              <1>       ;
3609 0000136B F6C308              <1>       test   bl, ALT_SHIFT       ; ARE WE IN ALTERNATE SHIFT
3610 0000136E 740E                <1>       jz     short K28A          ; IF NOT ALTERNATE
3611                              <1>       ; 12/04/2021
3612                              <1>       ;jz    K38
3613                              <1>       ;
3614 00001370 F6C710              <1>       test   bh, KBX             ; IS THIS THE ENCHANCED KEYBOARD?
3615 00001373 740E                <1>       jz     short K29           ; NO, ALT STATE IS REAL
3616                              <1>        ;28/02/2015
3617 00001375 F605[666E0000]04    <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; YES, IS SYSREQ KEY DOWN?
3618 0000137C 7405                <1>       jz     short K29           ;  NO, ALT STATE IS REAL
3619                              <1>       ; 12/04/2021
3620                              <1>       ;jnz   K38                 ; YES, THIS IS PHONY ALT STATE
3621                              <1>         ;                        ; DUE TO PRESSING SYSREQ
3622 0000137E E9C4000000          <1> K28A: jmp    K38
3623                              <1>       ;
3624                              <1>       ;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)
3625                              <1> K29:                               ; TEST-RESET
3626 00001383 F6C304              <1>       test   bl, CTL_SHIFT       ; ARE WE IN CONTROL SHIFT ALSO?
3627 00001386 740B                <1>       jz     short K31           ; NO_RESET
3628 00001388 3C53                <1>       cmp    al, DEL_KEY         ; CTL-ALT STATE, TEST FOR DELETE KEY
3629 0000138A 7507                <1>       jne    short K31           ; NO_RESET, IGNORE
3630                              <1>       ;
3631                              <1>       ;----- CTL-ALT-DEL HAS BEEN FOUND
3632                              <1>       ; 26/08/2014
3633                              <1> cpu_reset:
3634                              <1>       ; IBM PC/AT ROM BIOS source code - 10/06/85 (TEST4.ASM - PROC_SHUTDOWN)
3635                              <1>       ; Send FEh (system reset command) to the keyboard controller.
3636 0000138C B0FE                <1>       mov    al, SHUT_CMD        ; SHUTDOWN COMMAND
3637 0000138E E664                <1>       out    STATUS_PORT, al     ; SEND TO KEYBOARD CONTROL PORT
3638                              <1> khere:
3639 00001390 F4                  <1>       hlt                        ; WAIT FOR 80286 RESET
3640 00001391 EBFD                <1>       jmp    short khere         ; INSURE HALT
3641                              <1>       ;
3642                              <1>       ;----- IN ALTERNATE SHIFT, RESET NOT FOUND
3643                              <1> K31:                               ; NO-RESET
3644 00001393 3C39                <1>       cmp    al, 57              ; TEST FOR SPACE KEY
3645 00001395 7507                <1>       jne    short K311          ; NOT THERE
3646 00001397 B020                <1>       mov    al, ' '             ; SET SPACE CHAR
3647                              <1> k31a: ; 12/04/2021
3648 00001399 E929020000          <1>       jmp   K57                  ; BUFFER_FILL
3649                              <1> K311:
3650 0000139E 3C0F                <1>       cmp    al, 15              ; TEST FOR TAB KEY
3651 000013A0 7506                <1>       jne    short K312          ; NOT THERE
3652 000013A2 66B800A5            <1>       mov    ax, 0A500h          ; SET SPECIAL CODE FOR ALT-TAB
3653                              <1>       ;jmp   K57                 ; BUFFER_FILL
3654                              <1>       ; 12/04/2021
3655 000013A6 EBF1                <1>       jmp    short k31a
3656                              <1> K312:
3657 000013A8 3C4A                <1>       cmp    al, 74              ; TEST FOR KEY PAD -
3658                              <1>       ;je    short K37B          ; GO PROCESS
3659                              <1>       ; 12/04/2021
3660 000013AA 7404                <1>       je     short k312a
3661 000013AC 3C4E                <1>       cmp    al, 78              ; TEST FOR KEY PAD +
3662                              <1>         ;je  short K37B          ; GO PROCESS
3663                              <1>       ; 12/04/2021
3664 000013AE 7505                <1>       jne    short K32
3665                              <1> k312a:
3666 000013B0 E988000000          <1>       jmp    K37B
3667                              <1>       ;
3668                              <1>       ;----- LOOK FOR KEY PAD ENTRY
3669                              <1> K32:                               ; ALT-KEY-PAD
3670 000013B5 BF[286D0000]        <1>       mov    edi, K30            ; ALT-INPUT-TABLE offset
3671 000013BA B90A000000          <1>       mov    ecx, 10             ; LOOK FOR ENTRY USING KEYPAD
3672 000013BF F2AE                <1>       repne  scasb               ; LOOK FOR MATCH
3673 000013C1 7521                <1>       jne    short K33           ; NO_ALT_KEYPAD
3674 000013C3 F6C702              <1>       test   bh, LC_E0           ; IS THIS ONE OF THE NEW KEYS?
3675 000013C6 7579                <1>       jnz    short K37C          ; YES, JUMP, NOT NUMPAD KEY
3676 000013C8 81EF[296D0000]      <1>       sub    edi, K30+1          ; DI NOW HAS ENTRY VALUE
3677 000013CE A0[696E0000]        <1>       mov    al, [ALT_INPUT]     ; GET THE CURRENT BYTE
3678 000013D3 B40A                <1>       mov    ah, 10              ; MULTIPLY BY 10
3679 000013D5 F6E4                <1>       mul    ah
3680 000013D7 6601F8              <1>       add    ax, di              ; ADD IN THE LATEST ENTRY
3681 000013DA A2[696E0000]        <1>       mov    [ALT_INPUT], al     ; STORE IT AWAY
3682                              <1> K32A:
3683 000013DF E966FFFFFF          <1>       jmp     K26                ; THROW AWAY THAT KEYSTROKE
3684                              <1>       ;
3685                              <1>       ;----- LOOK FOR SUPERSHIFT ENTRY
3686                              <1> K33:                               ; NO-ALT-KEYPAD
3687 000013E4 C605[696E0000]00    <1>       mov   byte [ALT_INPUT], 0 ; ZERO ANY PREVIOUS ENTRY INTO INPUT
3688 000013EB B91A000000          <1>       mov    ecx, 26             ; (DI),(ES) ALREADY POINTING
3689 000013F0 F2AE                <1>       repne  scasb               ; LOOK FOR MATCH IN ALPHABET
```

```
3690 000013F2 7445                <1>       je    short K37A          ; MATCH FOUND, GO FILLL THE BUFFER
3691                              <1>       ;
3692                              <1>       ;----- LOOK FOR TOP ROW OF ALTERNATE SHIFT
3693                              <1> K34:                              ; ALT-TOP-ROW
3694 000013F4 3C02                <1>       cmp   al, 2                 ; KEY WITH '1' ON IT
3695 000013F6 7245                <1>       jb    short K37B          ; MUST BE ESCAPE
3696 000013F8 3C0D                <1>       cmp   al, 13                ; IS IT IN THE REGION
3697 000013FA 7705                <1>       ja    short K35           ; NO, ALT SOMETHING ELSE
3698 000013FC 80C476              <1>       add   ah, 118               ; CONVERT PSEUDO SCAN CODE TO RANGE
3699 000013FF EB38                <1>       jmp   short K37A          ; GO FILL THE BUFFER
3700                              <1>       ;
3701                              <1>       ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
3702                              <1> K35:                              ; ALT-FUNCTION
3703 00001401 3C57                <1>       cmp   al, F11_M             ; IS IT F11?
3704 00001403 7209                <1>       jb    short K35A ; 20/02/2015 ; NO, BRANCH
3705 00001405 3C58                <1>       cmp   al, F12_M             ; IS IT F12?
3706 00001407 7705                <1>       ja    short K35A ; 20/02/2015 ; NO, BRANCH
3707 00001409 80C434              <1>       add   ah, 52                ; CONVERT TO PSEUDO SCAN CODE
3708 0000140C EB2B                <1>       jmp   short K37A          ; GO FILL THE BUFFER
3709                              <1> K35A:
3710 0000140E F6C702              <1>       test  bh, LC_E0             ; DO WE HAVE ONE OF THE NEW KEYS?
3711 00001411 741B                <1>       jz    short K37           ; NO, JUMP
3712 00001413 3C1C                <1>       cmp   al, 28                ; TEST FOR KEYPAD ENTER
3713 00001415 7506                <1>       jne   short K35B          ; NOT THERE
3714 00001417 66B800A6            <1>       mov   ax, 0A600h          ; SPECIAL CODE
3715                              <1>       ;jmp  K57                 ; BUFFER FILL
3716                              <1>       ; 12/04/2021
3717 0000141B EB0C                <1>       jmp   short k35c
3718                              <1> K35B:
3719 0000141D 3C53                <1>       cmp   al, 83                ; TEST FOR DELETE KEY
3720 0000141F 7420                <1>       je    short K37C          ; HANDLE WITH OTHER EDIT KEYS
3721 00001421 3C35                <1>       cmp   al, 53                ; TEST FOR KEYPAD /
3722 00001423 75BA                <1>       jne   short K32A          ; NOT THERE, NO OTHER E0 SPECIALS
3723                              <1>       ; 12/04/2021
3724                              <1>       ;jne  K26
3725 00001425 66B800A4            <1>       mov   ax, 0A400h          ; SPECIAL CODE1
3726                              <1> k35c: ; 12/04/2021
3727 00001429 E999010000          <1>       jmp   K57                 ; BUFFER FILL
3728                              <1> K37:
3729 0000142E 3C3B                <1>       cmp   al, 59                ; TEST FOR FUNCTION KEYS (F1)
3730 00001430 720B                <1>       jb    short K37B          ; NO FN, HANDLE W/OTHER EXTENDED
3731 00001432 3C44                <1>       cmp   al, 68                ; IN KEYPAD REGION?
3732 00001434 77A9                <1>       ja    short K32A          ; IF SO, IGNORE
3733                              <1>       ; 12/04/2021
3734                              <1>       ;ja   K26
3735 00001436 80C42D              <1>       add   ah, 45                ; CONVERT TO PSEUDO SCAN CODE
3736                              <1> K37A:
3737 00001439 B000                <1>       mov   al, 0                 ; ASCII CODE OF ZERO
3738                              <1>       ;jmp  K57                 ; PUT IT IN THE BUFFER
3739                              <1>       ; 12/04/2021
3740 0000143B EBEC                <1>       jmp   short k35c
3741                              <1> K37B:
3742 0000143D B0F0                <1>       mov   al, 0F0h              ; USE SPECIAL ASCII CODE
3743                              <1>       ;jmp  K57                 ; PUT IT IN THE BUFFER
3744                              <1>       ; 12/04/2021
3745 0000143F EBE8                <1>       jmp   short k35c
3746                              <1> K37C:
3747 00001441 0450                <1>       add   al, 80                ; CONVERT SCAN CODE (EDIT KEYS)
3748 00001443 88C4                <1>       mov   ah, al                ; (SCAN CODE NOT IN AH FOR INSERT)
3749 00001445 EBF2                <1>       jmp   short K37A          ; PUT IT IN THE BUFFER
3750                              <1>       ;
3751                              <1>       ;----- NOT IN ALTERNATE SHIFT
3752                              <1> K38:                              ; NOT-ALT-SHIFT
3753                              <1>                                   ; BL STILL HAS SHIFT FLAGS
3754 00001447 F6C304              <1>       test  bl, CTL_SHIFT         ; ARE WE IN CONTROL SHIFT?
3755                              <1>       ;;jnz  short K38A         ; YES, START PROCESSING
3756                              <1>       ;jz   K44                 ; NOT-CTL-SHIFT
3757                              <1>       ; 12/04/2021
3758 0000144A 7505                <1>       jnz   short K38A          ; YES, START PROCESSING
3759 0000144C E9AB000000          <1>       jmp   K44                 ; NOT-CTL-SHIFT
3760                              <1>       ;
3761                              <1>       ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
3762                              <1>       ;----- TEST FOR BREAK
3763                              <1> K38A:
3764 00001451 3C46                <1>       cmp   al, SCROLL_KEY        ; TEST FOR BREAK
3765 00001453 7530                <1>       jne   short K39           ; JUMP, NO-BREAK
3766 00001455 F6C710              <1>       test  bh, KBX               ; IS THIS THE ENHANCED KEYBOARD?
3767 00001458 7405                <1>       jz    short K38B          ; NO, BREAK IS VALID
3768 0000145A F6C702              <1>       test  bh, LC_E0             ; YES, WAS LAST CODE AN E0?
3769 0000145D 7426                <1>       jz    short K39           ; NO-BREAK, TEST FOR PAUSE
3770                              <1> K38B:
3771 0000145F 8B1D[726E0000]      <1>       mov   ebx, [BUFFER_HEAD]  ; RESET BUFFER TO EMPTY
3772 00001465 891D[766E0000]      <1>       mov   [BUFFER_TAIL], ebx
3773 0000146B C605[646E0000]80    <1>       mov   byte [BIOS_BREAK], 80h  ; TURN ON BIOS_BREAK BIT
3774                              <1>       ;
3775                              <1>       ;----- ENABLE KEYBOARD
3776 00001472 B0AE                <1>       mov   al, ENA_KBD         ; ENABLE KEYBOARD
3777 00001474 E8B7010000          <1>       call  SHIP_IT             ; EXECUTE ENABLE
3778                              <1>       ;
3779                              <1>       ; CTRL+BREAK code here !!!
3780                              <1>       ;INT  1BH                 ; BREAK INTERRUPT VECTOR
3781                              <1>       ; 17/10/2015
3782 00001479 E832600000          <1>       call  ctrlbrk ; control+break subroutine
3783                              <1>       ;
3784                              <1>       ;sub  ax, ax              ; PUT OUT DUMMY CHARACTER
3785                              <1>       ; 12/04/2021
3786 0000147E 29C0                <1>       sub   eax, eax
3787 00001480 E942010000          <1>       jmp   K57                 ; BUFFER_FILL
3788                              <1>       ;
3789                              <1>       ;----- TEST FOR PAUSE
3790                              <1> K39:                              ; NO_BREAK
3791 00001485 F6C710              <1>       test  bh, KBX               ; IS THIS THE ENHANCED KEYBOARD?
3792 00001488 7537                <1>       jnz   short K41           ; YES, THEN THIS CAN'T BE PAUSE
3793 0000148A 3C45                <1>       cmp   al, NUM_KEY         ; LOOK FOR PAUSE KEY
3794 0000148C 7533                <1>       jne   short K41           ; NO-PAUSE
```

```
                                <1> K39P:
0000148E 800D[666E0000]08       <1>       or    byte [KB_FLAG_1], HOLD_STATE ; TURN ON THE HOLD FLAG
                                <1>       ;
                                <1>       ;----- ENABLE KEYBOARD
00001495 B0AE                   <1>       mov   al, ENA_KBD         ; ENABLE KEYBOARD
00001497 E894010000             <1>       call  SHIP_IT             ; EXECUTE ENABLE
                                <1> K39A:
0000149C B020                   <1>       mov   al, EOI             ; END OF INTERRUPT TO CONTROL PORT
0000149E E620                   <1>       out   20h, al ;out INTA00, al ; ALLOW FURTHER KEYSTROKE INTERRUPTS
                                <1>       ;
                                <1>       ;----- DURING PAUSE INTERVAL, TURN COLOR CRT BACK ON
000014A0 803D[9A6E0000]07       <1>       cmp   byte [CRT_MODE], 7  ; IS THIS BLACK AND WHITE CARD
000014A7 740A                   <1>       je    short K40           ; YES, NOTHING TO DO
000014A9 66BAD803               <1>       mov   dx, 03D8h           ; PORT FOR COLOR CARD
000014AD A0[9B6E0000]           <1>       mov   al, [CRT_MODE_SET]  ; GET THE VALUE OF THE CURRENT MODE
000014B2 EE                     <1>       out   dx, al              ; SET THE CRT MODE, SO THAT CRT IS ON
                                <1>       ;
                                <1> K40:                            ; PAUSE-LOOP
000014B3 F605[666E0000]08       <1>       test  byte [KB_FLAG_1], HOLD_STATE ; CHECK HOLD STATE FLAG
000014BA 75F7                   <1>       jnz   short K40           ; LOOP UNTIL FLAG TURNED OFF
                                <1>       ;
000014BC E995FEFFFF             <1>       jmp     K27               ; INTERRUPT_RETURN_NO_EOI
                                <1>         ;
                                <1>       ;----- TEST SPECIAL CASE KEY 55
                                <1> K41:                            ; NO-PAUSE
000014C1 3C37                   <1>       cmp   al, 55              ; TEST FOR */PRTSC KEY
000014C3 7513                   <1>       jne   short K42           ; NOT-KEY-55
000014C5 F6C710                 <1>       test  bh, KBX             ; IS THIS THE ENHANCED KEYBOARD?
000014C8 7405                   <1>       jz    short K41A          ; NO, CTL-PRTSC IS VALID
000014CA F6C702                 <1>       test  bh, LC_E0           ; YES, WAS LAST CODE AN E0?
000014CD 7421                   <1>       jz    short K42B          ; NO, TRANSLATE TO A FUNCTION
                                <1> K41A:
000014CF 66B80072               <1>       mov   ax, 114*256         ; START/STOP PRINTING SWITCH
000014D3 E9EF000000             <1>       jmp   K57                 ; BUFFER_FILL
                                <1>       ;
                                <1>       ;----- SET UP TO TRANSLATE CONTROL SHIFT
                                <1> K42:                            ; NOT-KEY-55
000014D8 3C0F                   <1>       cmp   al, 15              ; IS IT THE TAB KEY?
000014DA 7414                   <1>       je    short K42B          ; YES, XLATE TO FUNCTION CODE
000014DC 3C35                   <1>       cmp   al, 53              ; IS IT THE / KEY?
000014DE 750E                   <1>       jne   short K42A          ; NO, NO MORE SPECIAL CASES
000014E0 F6C702                 <1>       test  bh, LC_E0           ; YES, IS IT FROM THE KEY PAD?
000014E3 7409                   <1>       jz    short K42A          ; NO, JUST TRANSLATE
000014E5 66B80095               <1>       mov   ax, 9500h           ; YES, SPECIAL CODE FOR THIS ONE
000014E9 E9D9000000             <1>       jmp   K57                 ; BUFFER FILL
                                <1> K42A:
                                <1>       ;;mov  ebx, _K8           ; SET UP TO TRANSLATE CTL
000014EE 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:
000014F0 BB[5C6D0000]           <1>       mov   ebx, _K8            ; SET UP TO TRANSLATE CTL
                                <1>       ;jb   K56 ;; 20/02/2015
                                <1>       ; 12/04/2021
000014F5 7267                   <1>       jb    short K45F
000014F7 E9B9000000             <1>       jmp   K64
                                <1>         ;
                                <1>       ;----- NOT IN CONTROL SHIFT
                                <1> K44:                            ; NOT-CTL-SHIFT
000014FC 3C37                   <1>       cmp   al, 55              ; PRINT SCREEN KEY?
000014FE 7528                   <1>       jne   short K45           ; NOT PRINT SCREEN
00001500 F6C710                 <1>       test  bh, KBX             ; IS THIS ENHANCED KEYBOARD?
00001503 7407                   <1>       jz    short K44A          ; NO, TEST FOR SHIFT STATE
00001505 F6C702                 <1>       test  bh, LC_E0           ; YES, LAST CODE A MARKER?
00001508 7507                   <1>       jnz   short K44B          ; YES, IS PRINT SCREEN
0000150A EB41                   <1>       jmp   short K45C          ; NO, TRANSLATE TO '*' CHARACTER
                                <1> K44A:
0000150C F6C303                 <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT ; NOT 101 KBD, SHIFT KEY DOWN?
0000150F 743C                   <1>       jz    short K45C          ; NO, TRANSLATE TO '*' CHARACTER
                                <1>       ;
                                <1>       ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
                                <1> K44B:
00001511 B0AE                   <1>       mov   al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
00001513 E818010000             <1>       call  SHIP_IT             ; EXECUTE ENABLE
00001518 B020                   <1>       mov   al, EOI             ; END OF CURRENT INTERRUPT
0000151A 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
0000151C 8025[686E0000]FC       <1>       and   byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; ZERO OUT THESE FLAGS
00001523 E92EFEFFFF             <1>       jmp   K27                 ; GO BACK WITHOUT EOI OCCURRING
                                <1>       ;
                                <1>       ;----- HANDLE IN-CORE KEYS
                                <1> K45:                            ; NOT-PRINT-SCREEN
00001528 3C3A                   <1>       cmp   al, 58              ; TEST FOR IN-CORE AREA
0000152A 7734                   <1>       ja    short K46           ; JUMP IF NOT
0000152C 3C35                   <1>       cmp   al, 53              ; IS THIS THE '/' KEY?
0000152E 7505                   <1>       jne   short K45A          ; NO, JUMP
00001530 F6C702                 <1>       test  bh, LC_E0           ; WAS THE LAST CODE THE MARKER?
00001533 7518                   <1>       jnz   short K45C          ; YES, TRANSLATE TO CHARACTER
                                <1> K45A:
00001535 B91A000000             <1>       mov   ecx, 26             ; LENGHT OF SEARCH
0000153A BF[326D0000]           <1>       mov   edi, K30+10         ; POINT TO TABLE OF A-Z CHARS
0000153F F2AE                   <1>       repne scasb               ; IS THIS A LETTER KEY?
                                <1>       ; 20/02/2015
00001541 7505                   <1>       jne   short K45B          ; NO, SYMBOL KEY
                                <1>       ;
00001543 F6C340                 <1>       test  bl, CAPS_STATE      ; ARE WE IN CAPS_LOCK?
00001546 750C                   <1>       jnz   short K45D          ; TEST FOR SURE
                                <1> K45B:
00001548 F6C303                 <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
0000154B 750C                   <1>       jnz   short K45E          ; YES, UPPERCASE
                                <1>                                 ; NO, LOWERCASE
```

```
3900                                    <1> K45C:
3901 0000154D BB[B46D0000]              <1>         mov    ebx, K10             ; TRANSLATE TO LOWERCASE LETTERS
3902 00001552 EB51                      <1>         jmp    short K56
3903                                    <1> K45D:                               ; ALMOST-CAPS-STATE
3904 00001554 F6C303                    <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; CL ON. IS SHIFT ON, TOO?
3905 00001557 75F4                      <1>         jnz    short K45C           ; SHIFTED TEMP OUT OF CAPS STATE
3906                                    <1> K45E:
3907 00001559 BB[0C6E0000]              <1>         mov    ebx, K11             ; TRANSLATE TO UPPER CASE LETTERS
3908 0000155E EB45                      <1> K45F:   jmp    short K56
3909                                    <1>         ;
3910                                    <1>         ;----- TEST FOR KEYS F1 - F10
3911                                    <1> K46:                                ; NOT IN-CORE AREA
3912 00001560 3C44                      <1>         cmp    al, 68               ; TEST FOR F1 - F10
3913                                    <1>         ;ja    short K47            ; JUMP IF NOT
3914                                    <1>         ;jmp   short K53            ; YES, GO DO FN KEY PROCESS
3915 00001562 7635                      <1>         jna    short K53
3916                                    <1>         ;
3917                                    <1>         ;----- HANDLE THE NUMERIC PAD KEYS
3918                                    <1> K47:                                ; NOT F1 - F10
3919 00001564 3C53                      <1>         cmp    al, 83               ; TEST NUMPAD KEYS
3920 00001566 772D                      <1>         ja     short K52            ; JUMP IF NOT
3921                                    <1>         ;
3922                                    <1>         ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
3923                                    <1> K48:
3924 00001568 3C4A                      <1>         cmp    al, 74               ; SPECIAL CASE FOR MINUS
3925 0000156A 74ED                      <1>         je     short K45E           ; GO TRANSLATE
3926 0000156C 3C4E                      <1>         cmp    al, 78               ; SPECIAL CASE FOR PLUS
3927 0000156E 74E9                      <1>         je     short K45E           ; GO TRANSLATE
3928 00001570 F6C702                    <1>         test   bh, LC_E0            ; IS THIS ONE OFTHE NEW KEYS?
3929 00001573 750A                      <1>         jnz    short K49            ; YES, TRANSLATE TO BASE STATE
3930                                    <1>         ;
3931 00001575 F6C320                    <1>         test   bl, NUM_STATE        ; ARE WE IN NUM LOCK
3932 00001578 7514                      <1>         jnz    short K50            ; TEST FOR SURE
3933 0000157A F6C303                    <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
3934                                    <1>         ;jnz   short K51            ; IF SHIFTED, REALLY NUM STATE
3935 0000157D 75DA                      <1>         jnz    short K45E
3936                                    <1>         ;
3937                                    <1>         ;----- BASE CASE FOR KEYPAD
3938                                    <1> K49:
3939 0000157F 3C4C                      <1>         cmp    al, 76               ; SPECIAL CASE FOR BASE STATE 5
3940 00001581 7504                      <1>         jne    short K49A           ; CONTINUE IF NOT KEYPAD 5
3941 00001583 B0F0                      <1>         mov    al, 0F0h             ; SPECIAL ASCII CODE
3942 00001585 EB40                      <1>         jmp    short K57            ; BUFFER FILL
3943                                    <1> K49A:
3944 00001587 BB[B46D0000]              <1>         mov    ebx, K10             ; BASE CASE TABLE
3945 0000158C EB27                      <1>         jmp    short K64            ; CONVERT TO PSEUDO SCAN
3946                                    <1>         ;
3947                                    <1>         ;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS
3948                                    <1> K50:                                ; ALMOST-NUM-STATE
3949 0000158E F6C303                    <1>           test    bl, LEFT_SHIFT+RIGHT_SHIFT
3950 00001591 75EC                      <1>         jnz    short K49            ; SHIFTED TEMP OUT OF NUM STATE
3951 00001593 EBC4                      <1> K51:    jmp    short K45E           ; REALLY NUM STATE
3952                                    <1>         ;
3953                                    <1>         ;----- TEST FOR THE NEW KEYS ON WT KEYBOARDS
3954                                    <1> K52:                                ; NOT A NUMPAD KEY
3955 00001595 3C56                      <1>         cmp    al, 86               ; IS IT THE NEW WT KEY?
3956                                    <1>         ;jne   short K53            ; JUMP IF NOT
3957                                    <1>         ;jmp   short K45B           ; HANDLE WITH REST OF LETTER KEYS
3958 00001597 74AF                      <1>         je     short K45B
3959                                    <1>         ;
3960                                    <1>         ;----- MUST BE F11 OR F12
3961                                    <1> K53:                                ; F1 - F10 COME HERE, TOO
3962 00001599 F6C303                    <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST SHIFT STATE
3963 0000159C 74E1                      <1>         jz     short K49            ; JUMP, LOWER CASE PSEUDO SC'S
3964                                    <1>                ; 20/02/2015
3965 0000159E BB[0C6E0000]              <1>         mov    ebx, K11             ; UPPER CASE PSEUDO SCAN CODES
3966 000015A3 EB10                      <1>         jmp    short K64            ; TRANSLATE SCAN
3967                                    <1>         ;
3968                                    <1>         ;----- TRANSLATE THE CHARACTER
3969                                    <1> K56:                                ; TRANSLATE-CHAR
3970 000015A5 FEC8                      <1>         dec    al                   ; CONVERT ORIGIN
3971 000015A7 D7                        <1>         xlat                        ; CONVERT THE SCAN CODE TO ASCII
3972 000015A8 F605[686E0000]02          <1>         test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
3973 000015AF 7416                      <1>         jz     short K57            ; NO, GO FILL BUFFER
3974 000015B1 B4E0                      <1>         mov    ah, MC_E0            ; YES, PUT SPECIAL MARKER IN AH
3975 000015B3 EB12                      <1>         jmp    short K57            ; PUT IT INTO THE BUFFER
3976                                    <1>         ;
3977                                    <1>         ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES
3978                                    <1> K64:                                ; TRANSLATE-SCAN-ORGD
3979 000015B5 FEC8                      <1>         dec    al                   ; CONVERT ORIGIN
3980 000015B7 D7                        <1>                xlat                         ; CTL TABLE SCAN
3981 000015B8 88C4                      <1>         mov    ah, al               ; PUT VALUE INTO AH
3982 000015BA B000                      <1>         mov    al, 0                ; ZERO ASCII CODE
3983 000015BC F605[686E0000]02          <1>         test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
3984 000015C3 7402                      <1>         jz     short K57            ; NO, GO FILL BUFFER
3985 000015C5 B0E0                      <1>         mov    al, MC_E0            ; YES, PUT SPECIAL MARKER IN AL
3986                                    <1>         ;
3987                                    <1>         ;----- PUT CHARACTER INTO BUFFER
3988                                    <1> K57:                                ; BUFFER_FILL
3989 000015C7 3CFF                      <1>         cmp    al, -1               ; IS THIS AN IGNORE CHAR
3990 000015C9 7405                      <1>         je     short K59            ; YES, DO NOTHING WITH IT
3991                                    <1>         ;je    K26                  ; YES, DO NOTHING WITH IT
3992 000015CB 80FCFF                    <1>         cmp    ah, -1               ; LOOK FOR -1 PSEUDO SCAN
3993                                    <1>         ;;jne  short K61            ; NEAR_INTERRUPT_RETURN
3994                                    <1>         ;je    K26                  ; INTERRUPT_RETURN
3995                                    <1>         ; 12/04/2021
3996 000015CE 7505                      <1>         jne    short _K60           ; NEAR_INTERRUPT_RETURN
3997                                    <1> K59:                                ; NEAR_INTERRUPT_RETURN
3998 000015D0 E975FDFFFF                <1>         jmp    K26                  ; INTERRUPT_RETURN
3999                                    <1>
4000                                    <1> _K60: ; 29/01/2016
4001 000015D5 80FC68                    <1>         cmp    ah, 68h ; ALT + F1 key
4002 000015D8 721F                      <1>         jb     short K61
4003 000015DA 80FC6F                    <1>         cmp    ah, 6Fh ; ALT + F8 key
4004 000015DD 771A                      <1>         ja     short K61
```

```
                                <1>       ;
000015DF 8A1D[1E890100]         <1>       mov   bl, [ACTIVE_PAGE]
000015E5 80C368                 <1>       add   bl, 68h
000015E8 38E3                   <1>       cmp   bl, ah
000015EA 740D                   <1>       je    short K61
000015EC 6650                   <1>       push  ax
000015EE 88E0                   <1>       mov   al, ah
000015F0 2C68                   <1>       sub   al, 68h
000015F2 E84E090000             <1>       call  set_active_page
000015F7 6658                   <1>       pop   ax
                                <1> K61:                          ; NOT-CAPS-STATE
000015F9 8B1D[766E0000]         <1>       mov   ebx, [BUFFER_TAIL]  ; GET THE END POINTER TO THE BUFFER
000015FF 89DE                   <1>       mov   esi, ebx            ; SAVE THE VALUE
00001601 E8AFFAFFFF             <1>       call  _K4                 ; ADVANCE THE TAIL
00001606 3B1D[726E0000]         <1>       cmp   ebx, [BUFFER_HEAD]  ; HAS THE BUFFER WRAPPED AROUND
0000160C 740E                   <1>       je    short K62           ; BUFFER_FULL_BEEP
0000160E 668906                 <1>       mov   [esi], ax           ; STORE THE VALUE
00001611 891D[766E0000]         <1>       mov   [BUFFER_TAIL], ebx  ; MOVE THE POINTER UP
00001617 E92EFDFFFF             <1>       jmp   K26
                                <1>       ;;cli                     ; TURN OFF INTERRUPTS
                                <1>       ;;mov  al, EOI            ; END OF INTERRUPT COMMAND
                                <1>       ;;out  INTA00, al         ; SEND COMMAND TO INTERRUPT CONTROL PORT
                                <1>       ;MOV   AL, ENA_KBD        ; INSURE KEYBOARD IS ENABLED
                                <1>       ;CALL  SHIP_IT            ; EXECUTE ENABLE
                                <1>       ;MOV   AX, 9102H          ; MOVE IN POST CODE & TYPE
                                <1>       ;INT   15H                ; PERFORM OTHER FUNCTION
                                <1>       ;;and  byte [KB_FLAG_3],~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
                                <1>       ;JMP   K27A               ; INTERRUPT_RETURN
                                <1>       ;;jmp   K27
                                <1>       ;
                                <1>       ;----- BUFFER IS FULL SOUND THE BEEPER
                                <1> K62:
0000161C B020                   <1>       mov   al, EOI             ; ENABLE INTERRUPT CONTROLLER CHIP
0000161E E620                   <1>       out   INTA00, al
00001620 66B9A602               <1>       mov   cx, 678             ; DIVISOR FOR 1760 HZ
00001624 B304                   <1>       mov   bl, 4               ; SHORT BEEP COUNT (1/16 + 1/64 DELAY)
00001626 E8500D0000             <1>       call  beep                ; GO TO COMMON BEEP HANDLER
0000162B E926FDFFFF             <1>       jmp    K27                ; EXIT
                                <1>
                                <1> SHIP_IT:
                                <1>       ;--------------------------------------------------------------------------------
                                <1>       ; SHIP_IT
                                <1>       ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
                                <1>       ;      TO THE KEYBOARD CONTROLLER.
                                <1>       ;--------------------------------------------------------------------------------
                                <1>       ;
                                <1>
                                <1>       ;push  ax                 ; SAVE DATA TO SEND
                                <1>       ; 12/04/2021
00001630 50                     <1>       push  eax
                                <1>
                                <1>       ;----- WAIT FOR COMMAND TO ACCEPTED
00001631 FA                     <1>       cli                       ; DISABLE INTERRUPTS TILL DATA SENT
                                <1>       ; xor  ecx, ecx           ; CLEAR TIMEOUT COUNTER
00001632 B900000100             <1>       mov   ecx, 10000h
                                <1> S10:
00001637 E464                   <1>       in    al, STATUS_PORT     ; READ KEYBOARD CONTROLLER STATUS
00001639 A802                   <1>       test  al, INPT_BUF_FULL   ; CHECK FOR ITS INPUT BUFFER BUSY
0000163B E0FA                   <1>       loopnz S10                ; WAIT FOR COMMAND TO BE ACCEPTED
                                <1>
                                <1>       ;pop  ax                  ; GET DATA TO SEND
                                <1>       ; 12/04/2021
0000163D 58                     <1>       pop   eax
                                <1>
0000163E E664                   <1>       out   STATUS_PORT, al     ; SEND TO KEYBOARD CONTROLLER
00001640 FB                     <1>       sti                       ; ENABLE INTERRUPTS AGAIN
00001641 C3                     <1>       retn                      ; RETURN TO CALLER
                                <1>
                                <1>       ; 12/04/2021 (32 bit push/pop)
                                <1> SND_DATA:
                                <1>       ; -------------------------------------------------------------------------------
                                <1>       ; SND_DATA
                                <1>       ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
                                <1>       ;      TO THE KEYBOARD AND RECEIPT OF ACKNOWLEDGEMENTS. IT ALSO
                                <1>       ;      HANDLES ANY RETRIES IF REQUIRED
                                <1>       ; -------------------------------------------------------------------------------
                                <1>       ;
00001642 50                     <1>       push  eax ; push ax       ; SAVE REGISTERS
00001643 53                     <1>       push  ebx ; push bx
00001644 51                     <1>       push  ecx
00001645 88C7                   <1>       mov   bh, al              ; SAVE TRANSMITTED BYTE FOR RETRIES
00001647 B303                   <1>       mov   bl, 3               ; LOAD RETRY COUNT
                                <1> SD0:
00001649 FA                     <1>       cli                       ; DISABLE INTERRUPTS
0000164A 8025[676E0000]CF       <1>       and   byte [KB_FLAG_2], ~(KB_FE+KB_FA) ; CLEAR ACK AND RESEND FLAGS
                                <1>       ;
                                <1>       ;----- WAIT FOR COMMAND TO BE ACCEPTED
00001651 B900000100             <1>       mov   ecx, 10000h         ; MAXIMUM WAIT COUNT
                                <1> SD5:
00001656 E464                   <1>       in    al, STATUS_PORT     ; READ KEYBOARD PROCESSOR STATUS PORT
00001658 A802                   <1>       test  al, INPT_BUF_FULL   ; CHECK FOR ANY PENDING COMMAND
0000165A E0FA                   <1>       loopnz SD5                ; WAIT FOR COMMAND TO BE ACCEPTED
                                <1>       ;
0000165C 88F8                   <1>       mov   al, bh              ; REESTABLISH BYTE TO TRANSMIT
0000165E E660                   <1>       out   PORT_A, al          ; SEND BYTE
00001660 FB                     <1>       sti                       ; ENABLE INTERRUPTS
                                <1>       ;mov  cx, 01A00h          ; LOAD COUNT FOR 10 ms+
00001661 B9FFFF0000             <1>       mov   ecx, 0FFFFh
                                <1> SD1:
00001666 F605[676E0000]30       <1>       test  byte [KB_FLAG_2], KB_FE+KB_FA ; SEE IF EITHER BIT SET
0000166D 750F                   <1>       jnz   short SD3           ; IF SET, SOMETHING RECEIVED GO PROCESS
0000166F E2F5                   <1>       loop  SD1                 ; OTHERWISE WAIT
                                <1> SD2:
00001671 FECB                   <1>       dec   bl                  ; DECREMENT RETRY COUNT
00001673 75D4                   <1>       jnz   short SD0           ; RETRY TRANSMISSION
```

```
4110 00001675 800D[676E0000]80    <1>       or    byte [KB_FLAG_2], KB_ERR ; TURN ON TRANSMIT ERROR FLAG
4111 0000167C EB09                <1>       jmp   short SD4          ; RETRIES EXHAUSTED FORGET TRANSMISSION
4112                              <1> SD3:
4113 0000167E F605[676E0000]10    <1>       test  byte [KB_FLAG_2], KB_FA ; SEE IF THIS IS AN ACKNOWLEDGE
4114 00001685 74EA                <1>       jz    short SD2          ; IF NOT, GO RESEND
4115                              <1> SD4:
4116 00001687 59                  <1>       pop   ecx                ; RESTORE REGISTERS
4117 00001688 5B                  <1>       pop   ebx ; pop bx
4118 00001689 58                  <1>       pop   eax ; pop ax
4119 0000168A C3                  <1>       retn                     ; RETURN, GOOD TRANSMISSION
4120                              <1>
4121                              <1> SND_LED:
4122                              <1>       ; ------------------------------------------------------------------------------
4123                              <1>       ; SND_LED
4124                              <1>       ;     THIS ROUTINES TURNS ON THE MODE INDICATORS.
4125                              <1>       ;
4126                              <1>       ;------------------------------------------------------------------------------
4127                              <1>       ;
4128 0000168B FA                  <1>       cli                      ; TURN OFF INTERRUPTS
4129 0000168C F605[676E0000]40    <1>       test  byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
4130 00001693 755F                <1>       jnz   short SL1          ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
4131                              <1>       ;
4132 00001695 800D[676E0000]40    <1>       or    byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
4133 0000169C B020                <1>       mov   al, EOI            ; END OF INTERRUPT COMMAND
4134 0000169E E620                <1>       out   20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
4135 000016A0 EB11                <1>       jmp   short SL0          ; GO SEND MODE INDICATOR COMMAND
4136                              <1> SND_LED1:
4137 000016A2 FA                  <1>       cli                      ; TURN OFF INTERRUPTS
4138 000016A3 F605[676E0000]40    <1>       test  byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
4139 000016AA 7548                <1>       jnz   short SL1          ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
4140                              <1>       ;
4141 000016AC 800D[676E0000]40    <1>       or    byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
4142                              <1> SL0:
4143 000016B3 B0ED                <1>       mov   al, LED_CMD        ; LED CMD BYTE
4144 000016B5 E888FFFFFF          <1>       call  SND_DATA           ; SEND DATA TO KEYBOARD
4145 000016BA FA                  <1>       cli
4146 000016BB E836000000          <1>       call  MAKE_LED           ; GO FORM INDICATOR DATA BYTE
4147 000016C0 8025[676E0000]F8    <1>       and   byte [KB_FLAG_2], 0F8h ; ~KB_LEDS ; CLEAR MODE INDICATOR BITS
4148 000016C7 0805[676E0000]      <1>       or    [KB_FLAG_2], al    ; SAVE PRESENT INDICATORS FOR NEXT TIME
4149 000016CD F605[676E0000]80    <1>       test  byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
4150 000016D4 750F                <1>       jnz   short SL2          ; IF SO, BYPASS SECOND BYTE TRANSMISSION
4151                              <1>       ;
4152 000016D6 E867FFFFFF          <1>       call  SND_DATA           ; SEND DATA TO KEYBOARD
4153 000016DB FA                  <1>       cli                      ; TURN OFF INTERRUPTS
4154 000016DC F605[676E0000]80    <1>       test  byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
4155 000016E3 7408                <1>       jz    short SL3          ; IF NOT, DON'T SEND AN ENABLE COMMAND
4156                              <1> SL2:
4157 000016E5 B0F4                <1>       mov   al, KB_ENABLE      ; GET KEYBOARD CSA ENABLE COMMAND
4158 000016E7 E856FFFFFF          <1>       call  SND_DATA           ; SEND DATA TO KEYBOARD
4159 000016EC FA                  <1>       cli                      ; TURN OFF INTERRUPTS
4160                              <1> SL3:
4161 000016ED 8025[676E0000]3F    <1>       and   byte [KB_FLAG_2], ~(KB_PR_LED+KB_ERR) ; TURN OFF MODE INDICATOR
4162                              <1> SL1:                           ; UPDATE AND TRANSMIT ERROR FLAG
4163 000016F4 FB                  <1>       sti                      ; ENABLE INTERRUPTS
4164 000016F5 C3                  <1>       retn                     ; RETURN TO CALLER
4165                              <1>
4166                              <1> MAKE_LED:
4167                              <1>       ;------------------------------------------------------------------------------
4168                              <1>       ; MAKE_LED
4169                              <1>       ;     THIS ROUTINES FORMS THE DATA BYTE NECESSARY TO TURN ON/OFF
4170                              <1>       ;     THE MODE INDICATORS.
4171                              <1>       ;------------------------------------------------------------------------------
4172                              <1>       ;
4173                              <1>       ;push  cx                ; SAVE CX
4174 000016F6 A0[656E0000]        <1>       mov   al, [KB_FLAG]      ; GET CAPS & NUM LOCK INDICATORS
4175 000016FB 2470                <1>       and   al, CAPS_STATE+NUM_STATE+SCROLL_STATE ; ISOLATE INDICATORS
4176                              <1>       ;mov  cl, 4              ; SHIFT COUNT
4177                              <1>       ;rol  al, cl             ; SHIFT BITS OVER TO TURN ON INDICATORS
4178 000016FD C0C004              <1>       rol   al, 4 ; 20/02/2015
4179 00001700 2407                <1>       and   al, 07h            ; MAKE SURE ONLY MODE BITS ON
4180                              <1>       ;pop  cx
4181 00001702 C3                  <1>       retn                     ; RETURN TO CALLER
4182                              <1>
4183                              <1> ; % include 'kybdata.s'   ; KEYBOARD DATA
4184                              <1>
4185                              <1>
4186                              <1> ; /// End Of KEYBOARD FUNCTIONS ///
2658
2659                                  %include 'video.s' ; 07/03/2015
2660                              <1> ; ****************************************************************************
2661                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - video.s
2662                              <1> ; ----------------------------------------------------------------------------
2663                              <1> ; Last Update: 12/04/2021
2664                              <1> ; ----------------------------------------------------------------------------
2665                              <1> ; Beginning: 16/01/2016
2666                              <1> ; ----------------------------------------------------------------------------
2667                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
2668                              <1> ; ----------------------------------------------------------------------------
2669                              <1> ; Turkish Rational DOS
2670                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
2671                              <1> ;
2672                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
2673                              <1> ; video.inc (13/08/2015)
2674                              <1> ;
2675                              <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
2676                              <1> ; ****************************************************************************
2677                              <1>
2678                              <1> ; Retro UNIX 386 v1 Kernel - VIDEO.INC
2679                              <1> ; Last Modification: 13/08/2015
2680                              <1> ;             (Video Data is in 'VIDATA.INC')
2681                              <1> ;
2682                              <1> ; ///////// VIDEO (CGA) FUNCTIONS ///////////////
2683                              <1>
2684                              <1> ; 16/01/2016 (32 bit modifications, TRDOS386 - TRDOS v2.0, video.s)
2685                              <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)
00001703 9C                     <1>       pushfd
00001704 0E                     <1>       push   cs
00001705 E851000000             <1>       call   VIDEO_IO_1
0000170A 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>
0000170B [BC1A0000]     <1> M1:   dd    SET_MODE     ; TABLE OF ROUTINES WITHIN VIDEO I/O
0000170F [851E0000]     <1>       dd    SET_CTYPE
00001713 [B91E0000]     <1>       dd    SET_CPOS
00001717 [E11E0000]     <1>       dd    READ_CURSOR
                        <1>       ;dd   VIDEO_RETURN ; READ_LPEN
0000171B [BA1A0000]     <1>       dd    set_mode_ncm ; Set mode without clearing video memory
0000171F [271F0000]     <1>       dd    ACT_DISP_PAGE
00001723 [B91F0000]     <1>       dd    SCROLL_UP
00001727 [D9200000]     <1>       dd    SCROLL_DOWN
0000172B [58210000]     <1>       dd    READ_AC_CURRENT
0000172F [B5210000]     <1>       dd    WRITE_AC_CURRENT
00001733 [DB210000]     <1>       dd    WRITE_C_CURRENT
00001737 [0C2B0000]     <1>       dd    SET_COLOR
0000173B [772B0000]     <1>       dd    WRITE_DOT
0000173F [422B0000]     <1>       dd    READ_DOT
00001743 [65220000]     <1>       dd    WRITE_TTY
00001747 [A21A0000]     <1>       dd    VIDEO_STATE
0000174B [FF350000]     <1>       dd    vga_pal_funcs; 10/08/2016 (TRDOS 386)
0000174F [78300000]     <1>       dd    font_setup   ; 10/07/2016 (TRDOS 386)
00001753 [F11A0000]     <1>       dd    VIDEO_RETURN ; RESERVED
00001757 [DC230000]     <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
0000175B 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
```

```
2896 0000175C 80FC14              <1>       cmp    ah, M1L/4
2897 0000175F 7205                <1>       jb     short VGA_func
2898                              <1>
2899 00001761 80FC4F              <1>       cmp    ah, 4Fh
2900 00001764 7532                <1>       jne    short M4 ; invalid !
2901                              <1>
2902                              <1> VGA_func: ; 26/11/2020
2903 00001766 06                  <1>       push   es ; *
2904 00001767 1E                  <1>       push   ds ; **             ; SAVE WORK AND PARAMETER REGISTERS
2905                              <1>
2906                              <1>       ; 26/11/2020
2907 00001768 50                  <1>       push   eax ; -
2908                              <1>
2909 00001769 66B81000            <1>       mov    ax, KDATA           ; POINT DS: TO DATA SEGMENT
2910 0000176D 8ED8                <1>       mov    ds, ax
2911 0000176F 8EC0                <1>       mov    es, ax
2912                              <1>
2913                              <1>       ; 26/11/2020
2914 00001771 58                  <1>       pop    eax ; +
2915                              <1>       ;
2916 00001772 FB                  <1>       sti
2917 00001773 80FC4F              <1>       cmp    ah, 4Fh
2918 00001776 747A                <1>       je     short VBE_func
2919                              <1>
2920                              <1>       ; 04/12/2020
2921 00001778 A3[78950100]        <1>       mov    [video_eax], eax
2922                              <1>
2923                              <1>       ; 21/12/2020
2924 0000177D 803D[9A6E0000]FF    <1>       cmp    byte [CRT_MODE], 0FFh ; Current mode is a VESA VBE mode ?
2925 00001784 7213                <1>       jb     short VGA_func_std
2926                              <1>
2927 00001786 08E4                <1>       or     ah, ah              ; set mode ?
2928 00001788 750F                <1>       jnz    short VGA_func_std  ; no
2929                              <1>
2930 0000178A 803D[52090000]03    <1>       cmp    byte [vbe3], 3      ; (real) VESA VBE 3 video bios ?
2931 00001791 7506                <1>       jne    short VGA_func_std  ; no
2932                              <1>
2933 00001793 E989010000          <1>       jmp    vesa_vbe3_pmi
2934                              <1>
2935                              <1>       ; 21/12/2020
2936                              <1> M4:                               ; COMMAND NOT VALID
2937 00001798 CF                  <1>       iretd                       ; DO NOTHING IF NOT IN VALID RANGE
2938                              <1>
2939                              <1> VGA_func_std:
2940                              <1>       ; 06/12/2020
2941                              <1>       ; 03/12/2020
2942 00001799 80FC0F              <1>       cmp    ah, 0Fh
2943 0000179C 773D                <1>       ja     short VGA_funcs_0   ; only CGA funcs will be handled by
2944                              <1>       ; 06/12/2020               ; vga bios firmware
2945                              <1>       ; 03/12/2020
2946                              <1>       ;test  ah, 7Fh ; set mode ?
2947                              <1>       ;;or   ah, ah              ; only 'set mode' will be handled by
2948                              <1>       ;jnz   short VGA_funcs_0   ; vga bios firmware
2949                              <1>       ;jz    short vbe_pmi32_0
2950                              <1>
2951                              <1>       ; 28/11/2020
2952 0000179E 803D[1C120300]00    <1>       cmp    byte [pmi32], 0           ; 32 bit protected mode interface for
2953 000017A5 7634                <1>       jna    short VGA_funcs_0   ; video hardware's vga bios firmware
2954                              <1>                                  ; ([pmi32] > 0 if it is activated)
2955                              <1>                                  ; note:
2956                              <1>                                  ; [vbe3] = 3 is required to activate
2957                              <1>       ; 07/12/2020
2958 000017A7 20E4                <1>       and    ah, ah ; is this set mode ?
2959 000017A9 7413                <1>       jz     short vbe_pmi32_2 ; yes
2960                              <1>
2961 000017AB 80FC04              <1>       cmp    ah, 04h       ; set mode ('no clear memory' option)
2962 000017AE 742B                <1>       je     short VGA_funcs_0
2963                              <1>
2964                              <1>       ; 07/12/2020
2965 000017B0 803D[9A6E0000]07    <1>       cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
2966 000017B7 7622                <1>       jna    short VGA_funcs_0  ; no
2967                              <1>
2968                              <1>       ; when mode 3 is active,
2969                              <1>       ; video bios functions are not redirected
2970                              <1>       ; to VESA VBE3 PMI except 'set mode' function
2971                              <1>
2972                              <1> vbe_pmi32_0:
2973                              <1>       ; 06/12/2020
2974                              <1>       ;or    ah, ah
2975                              <1>       ;jnz   short vbe_pmi32_2
2976                              <1>       ; ah = 0 ; 'set mode'
2977                              <1>       ;cmp   al, 3 ; 80x25 text mode, 16 colors, default mode for MainProg
2978                              <1>       ;jne   short vbe_pmi32_1
2979                              <1>       ;cmp   byte [CRT_MODE], 3 ; If current video mode <> 3 and requested
2980                              <1>       ;                  ; video mode is 3, use internal 'set mode';
2981                              <1>       ;                  ; otherwise, use vesa vbe 3 bios's 'set mode'.
2982                              <1>       ;jne   short VGA_funcs_0
2983                              <1> vbe_pmi32_1:
2984 000017B9 E963010000          <1>       jmp    vesa_vbe3_pmi
2985                              <1> ;vbe_pmi32_2:
2986                              <1>       ;cmp   ah, 04h ; set mode (no clear mem option)
2987                              <1>       ;jne   short vbe_pmi32_1
2988                              <1>
2989                              <1> vbe_pmi32_2:
2990                              <1>       ; 07/12/2020
2991 000017BE 803D[9A6E0000]07    <1>       cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
2992 000017C5 77F2                <1>       ja     short vbe_pmi32_1  ; yes
2993                              <1>
2994 000017C7 3C07                <1>       cmp    al, 7  ; requested mode > 7 ?
2995 000017C9 7610                <1>       jna    short VGA_funcs_0  ; no (CGA)
2996                              <1>
2997 000017CB 3C13                <1>       cmp    al, 13h
2998 000017CD 76EA                <1>       jna    short vbe_pmi32_1
2999                              <1>
3000 000017CF A880                <1>       test   al, 80h
```

```
3001 000017D1 7408              <1>       jz     short VGA_funcs_0  ; unknown or special
3002                            <1>
3003 000017D3 3C87              <1>       cmp    al, 87h      ; requested mode > 7 ?
3004                            <1>                     ; (with no clear mem ops)
3005 000017D5 7604              <1>       jna    short VGA_funcs_0  ; no (CGA)
3006                            <1>
3007 000017D7 3C93              <1>       cmp    al, 93h
3008 000017D9 76DE              <1>       jna    short vbe_pmi32_1
3009                            <1>
3010                            <1>       ; > 13h video modes are unknown or special
3011                            <1>       ; they must be handled by kernel
3012                            <1>
3013                            <1>       ; CGA video modes will be handled by kernel
3014                            <1>
3015                            <1> VGA_funcs_0:
3016 000017DB 52                <1>       push   edx ; ***
3017 000017DC 51                <1>       push   ecx ; ****
3018 000017DD 53                <1>       push   ebx ; *****
3019 000017DE 56                <1>       push   esi ; ******
3020 000017DF 57                <1>       push   edi ; *******
3021 000017E0 55                <1>       push   ebp ; ********
3022                            <1>
3023                            <1>       ;mov   [video_eax], eax ; 12/05/2016
3024 000017E1 BF00800B00        <1>       mov    edi, 0B8000h        ; GET offset FOR COLOR CARD
3025                            <1>
3026                            <1>       ; 23/03/2016
3027 000017E6 C0E402            <1>       shl    ah, 2  ; dword            ; TIMES 2 FOR WORD TABLE LOOKUP
3028 000017E9 0FB6F4            <1>       movzx  esi, ah                   ; MOVE OFFSET INTO LOOK UP REGISTER (SI)
3029                            <1>       ;mov   ah, [CRT_MODE]            ; MOVE CURRENT MODE INTO (AH) REGISTER
3030                            <1>
3031                            <1>       ;;15/01/2017
3032                            <1>       ; 14/01/2017
3033                            <1>       ; 02/01/2017
3034                            <1>       ;;mov  byte [intflg], 31h  ; video interrupt
3035                            <1>       ;sti ; 26/11/2020
3036                            <1>       ;
3037                            <1>
3038 000017EC FFA6[0B170000]    <1>       JMP    dword [esi+M1]            ; GO TO SELECTED FUNCTION
3039                            <1>
3040                            <1> VBE_func:
3041                            <1>       ; 26/11/2020
3042                            <1>       ;sti
3043 000017F2 55                <1>       push   ebp ; *** ; 27/11/2020
3044 000017F3 56                <1>       push   esi ; ****
3045                            <1>
3046                            <1>       ; Note:
3047                            <1>       ; ebx, ecx, edx, edi, ebp registers
3048                            <1>       ; must be saved by VBE functions and
3049                            <1>       ; eax register must be set
3050                            <1>       ; (according to VBE 3 standard)
3051                            <1>       ; before return from this interrupt
3052                            <1>       ; (every function must restore and set
3053                            <1>       ; registers except esp, esi, es, ds)
3054                            <1>
3055 000017F4 803D[52090000]02  <1>       cmp    byte [vbe3], 2
3056 000017FB 7741              <1>       ja     short VESA_VBE3_PMI_CALL ; VBE3 video bios ('PMID')
3057                            <1>       ;je    short VBE_func_0  ; Bochs/Qemu/VirtualBox emulator
3058 000017FD 726A              <1>       jb     short VBE_unknown ; invalid ([vbe3] = 0)
3059                            <1>
3060                            <1>       ;jmp   VESA_VBE3_PMI_CALL
3061                            <1>
3062                            <1> VBE_func_0:
3063                            <1>       ; Bochs/Plex86 VGAbios VBE extension
3064                            <1>       ; (TRDOS 386 v2.0.3 can use VBE graphics modes on emulators)
3065                            <1>       ; BOCHS/QEMU/VIRTUALBOX
3066                            <1>
3067 000017FF 8A25[53090000]    <1>       mov    ah, [vbe2bios]
3068 00001805 80FCC0            <1>       cmp    ah, 0C0h
3069 00001808 725F              <1>       jb     short VBE_unknown
3070 0000180A 80FCC5            <1>       cmp    ah, 0C5h
3071 0000180D 775A              <1>       ja     short VBE_unknown
3072                            <1>
3073                            <1>       ; TRDOS 386 is running on BOCHS or QEMU
3074 0000180F B44F              <1>       mov    ah, 4Fh
3075                            <1> VBE_func_1:
3076 00001811 0FB6F0            <1>       movzx  esi, al ; VESA VBE function number
3077 00001814 66C1E602          <1>       shl    si, 2 ; dword
3078 00001818 6683FE14          <1>       cmp    si, N1L
3079 0000181C 734B              <1>       jnb    short VBE_unknown
3080                            <1>       ;sti
3081                            <1>
3082 0000181E FF96[2A180000]    <1>       call   dword [esi+N1] ; call VBE function
3083                            <1>
3084                            <1>       ;jmp   short VBE_bios_return
3085                            <1>
3086                            <1> VBE_bios_return:
3087 00001824 FA                <1>       cli
3088 00001825 5E                <1>       pop    esi ; ****
3089 00001826 5D                <1>       pop    ebp ; *** ; 27/11/2020
3090 00001827 07                <1>       pop    es  ; **
3091 00001828 1F                <1>       pop    ds  ; *
3092 00001829 CF                <1>       iretd
3093                            <1>
3094                            <1>       ; 26/11/2020
3095                            <1> N1:
3096 0000182A [8D180000]        <1>       dd     vbe_biosfn_return_ctrl_info
3097 0000182E [83390000]        <1>       dd     vbe_biosfn_return_mode_info
3098 00001832 [3E3A0000]        <1>       dd     vbe_biosfn_set_mode
3099 00001836 [0E3B0000]        <1>       dd     vbe_biosfn_return_current_mode
3100 0000183A [2D3B0000]        <1>       dd     vbe_biosfn_save_restore_state
3101                            <1>       ;dd    vbe_biosfn_display_window_ctrl
3102                            <1>       ;dd    vbe_biosfn_set_get_log_scanline
3103                            <1>       ;dd    vbe_biosfn_set_get_disp_start
3104                            <1>       ;dd    vbe_biosfn_set_get_dac_pal_frm
3105                            <1>       ;dd    vbe_biosfn_set_get_palette_data
```

```
3106                                    <1>
3107                                    <1> N1L   EQU   $ - N1
3108                                    <1>
3109                                    <1>
3110                                    <1> VESA_VBE3_PMI_CALL: ; VESA VBE video bios (firmware) functions
3111                                    <1>                     ; by using VESA VBE3 Protected Mode Interface
3112                                    <1>
3113                                    <1>       ; 29/11/2020
3114                                    <1>       ; 26/11/2020 - TRDOS 386 v2.0.3 video.s
3115                                    <1>
3116                                    <1>       ; We are here because..
3117                                    <1>       ; 'PMID' has been verified by TRDOS 386 v2.0.3 kernel.
3118                                    <1>       ; (Otherwise bochs/plex86 compatible VBE functions and
3119                                    <1>       ;  modes would be used on BOCHS/QEMU/VIRTUALBOX emulators
3120                                    <1>       ;  or only standard/old VGA graphics modes would be used.)
3121                                    <1>
3122                                    <1>       ; (TRDOS 386 v2.0.3 can use VESA VBE graphics modes if
3123                                    <1>       ;  the video bios is full compatible with VBE3 standard)
3124                                    <1>
3125                                    <1>       ; 29/11/2020
3126                                    <1>
3127 0000183E 0FB6F0                  <1>       movzx  esi, al ; VESA VBE 3 function number
3128 00001841 66C1E602                 <1>       shl    si, 2 ; dword
3129 00001845 6683FE14                 <1>       cmp    si, P1L
3130 00001849 731E                     <1>       jnb    short VBE_unknown
3131                                    <1>       ;sti
3132                                    <1>
3133 0000184B 57                       <1>       push   edi ; *****
3134                                    <1>
3135 0000184C FF96[55180000]           <1>       call   dword [esi+P1] ; call VBE 3 function
3136                                    <1>
3137 00001852 5F                       <1>       pop    edi ; *****
3138                                    <1>
3139 00001853 EBCF                     <1>       jmp    short VBE_bios_return
3140                                    <1>
3141                                    <1> P1:
3142 00001855 [6F180000]               <1>       dd     vbe3_pmfn_return_ctrl_info
3143 00001859 [93180000]               <1>       dd     vbe3_pmfn_return_mode_info
3144 0000185D [D0180000]               <1>       dd     vbe3_pmfn_set_mode
3145 00001861 [CB180000]               <1>       dd     vbe3_pmfn_return_current_mode
3146 00001865 [C2190000]               <1>       dd     vbe3_pmfn_save_restore_state
3147                                    <1>       ;dd    vbe3_pmfn_display_window_ctrl
3148                                    <1>       ;dd    vbe3_pmfn_set_get_log_scanline
3149                                    <1>       ;dd    vbe3_pmfn_set_get_disp_start
3150                                    <1>       ;dd    vbe3_pmfn_set_get_dac_pal_frm
3151                                    <1>       ;dd    vbe3_pmfn_set_get_palette_data
3152                                    <1>       ;dd    vbe3_pmfn_return_pmi ; invalid for TRDOS 386 v2
3153                                    <1>       ;dd    vbe3_pmfn_set_get_pixel_clock
3154                                    <1>
3155                                    <1> P1L   EQU   $ - P1
3156                                    <1>
3157                                    <1> ;     ; 29/11/2020
3158                                    <1> ;     mov    edi, VBE3MODEINFOBLOCK >> 4 ; / 16
3159                                    <1> ;
3160                                    <1> ;     cmp    al, 04h
3161                                    <1> ;     jb     short vbe3_pm_f ; function: 4F00h to 4F03h
3162                                    <1> ;     ja     short vbe3_pmi_f5B ; function: 4F05h to 4F0Bh
3163                                    <1> ;
3164                                    <1> ;     ; check buffer length (must be <= 2048 bytes)
3165                                    <1> ;
3166                                    <1> ;     and    dl, dl ; 0
3167                                    <1> ;     jz     short vbe3_pm_f
3168                                    <1> ;                  ; Return Save/Restore State buffer size
3169                                    <1> ;
3170                                    <1> ;     push   ebx  ; buffer address
3171                                    <1> ;     push   edx  ; function: save (01h) or restore (02h)
3172                                    <1> ;     call
3173                                    <1> ;
3174                                    <1> ;vbe3_pm_f03:
3175                                    <1> ;     cmp    al, 2
3176                                    <1> ;     ja     short vbe3_pm_f ; function 4F03h
3177                                    <1> ;     jb     short vbe3_pm_f1
3178                                    <1> ;
3179                                    <1> ;
3180                                    <1> ;vbe3_pm_f1:
3181                                    <1> ;
3182                                    <1> ;
3183                                    <1> ;vbe3_pmi_f5B:
3184                                    <1> ;     cmp    al, 09h
3185                                    <1> ;     jna    short vbe3_pm_f ; funcs 05h to 09h are usable
3186                                    <1> ;
3187                                    <1> ;     cmp    al, 0Bh ; Get/Set pixel clock, last function
3188                                    <1> ;     jne    short VBE_unknown
3189                                    <1> ;                  ; (do not use 'uncertain' functions
3190                                    <1> ;                  ; because of system-user buff transfers)
3191                                    <1> ;vbe3_pm_f:
3192                                    <1> ;     mov    byte [vbe3_pm_fn], al     ; set
3193                                    <1> ;     ; prepare 16 bit pm segments & registers for pmi call
3194                                    <1> ;     call   VESA_VBE3_PM_FUNCTION
3195                                    <1> ;
3196                                    <1> ;
3197                                    <1>
3198                                    <1>       ; 26/11/2020
3199                                    <1> VBE_unknown:
3200 00001869 66B80001                 <1>       mov    ax, 0100h  ; ah = 1 : Function call failed
3201                                    <1>                         ; al = 0 : Function is not supported
3202                                    <1>
3203 0000186D EBB5                     <1>       jmp    short VBE_bios_return
3204                                    <1>
3205                                    <1> vbe3_pmfn_return_ctrl_info:
3206                                    <1>       ; 12/12/2020
3207                                    <1>       ;
3208                                    <1>       ; VBE function 4F00h - Return VBE Controller Information
3209                                    <1>       ;
3210                                    <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>
0000186F 21FF           <1>       and    edi, edi
00001871 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:
00001873 57             <1>       push   edi
00001874 51             <1>       push   ecx
00001875 BE007E0900     <1>       mov    esi, VBE3INFOBLOCK
0000187A B900020000     <1>       mov    ecx, 512
0000187F E8A0010100     <1>       call   transfer_to_user_buffer
00001884 59             <1>       pop    ecx
00001885 5F             <1>       pop    edi
00001886 720F           <1>       jc     short vbe3_func_fail
                        <1>
00001888 31C0           <1>       xor    eax, eax
0000188A B04F           <1>       mov    al, 4Fh ; successful
                        <1> ;vbe_ctrl_info_retn:
0000188C 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>
0000188D 21FF           <1>       and    edi, edi
0000188F 7406           <1>       jz     short vbe3_func_fail ; invalid buffer addr !
00001891 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>
00001893 09FF                   <1>       or     edi, edi
00001895 7506                   <1>       jnz    short _vbe3_pmfn_return_mode_info
                                <1>
                                <1> vbe3_func_fail:
00001897 B84F010000             <1>       mov    eax, 014Fh ; ah = 1 : Function call failed
                                <1>                         ; al = 4Fh : Function is supported
0000189C C3                     <1>       retn
                                <1>
                                <1>       ; jump from '_vbe_biosfn_return_mode_info'
                                <1> _vbe3_pmfn_return_mode_info:
0000189D 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
0000189E BF00060000             <1>       mov    edi, VBE3MODEINFOBLOCK-VBE3SAVERESTOREBLOCK
                                <1>       ; ES selector base address = VBE3SAVERESTOREBLOCK
000018A3 E8FC000000             <1>       call   int10h_32bit_pmi
                                <1>
000018A8 5F                     <1>       pop    edi
                                <1>
                                <1>       ;cmp   ax, 004Fh
                                <1>       ;jne   short vbe3_func_retn  ; failed !
                                <1>
000018A9 21FF                   <1>       and    edi, edi
                                <1>       ;jz    short vbe3_func_success
                                <1>       ; 21/12/2020
000018AB 741D                   <1>       jz     short vbe3_func_retn
                                <1>
000018AD 6683F84F               <1>       cmp    ax, 004Fh
000018B1 7517                   <1>       jne    short vbe3_func_retn  ; failed !
                                <1>
000018B3 51                     <1>       push   ecx
000018B4 BE007C0900             <1>       mov    esi, VBE3MODEINFOBLOCK
000018B9 B900010000             <1>       mov    ecx, 256
000018BE E861010100             <1>       call   transfer_to_user_buffer
000018C3 59                     <1>       pop    ecx
000018C4 72D1                   <1>       jc     short vbe3_func_fail
                                <1>
000018C6 31C0                   <1>       xor    eax, eax
000018C8 B04F                   <1>       mov    al, 4Fh ; successful
                                <1> vbe3_func_success:
                                <1> vbe3_func_retn:
000018CA 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
000018CB 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
000018D0 57                     <1>       push   edi ; *****
000018D1 F6C708                 <1>       test   bh, 8  ; Use user specified CRTC values
000018D4 7530                   <1>       jnz    short vbe3_sm_3  ; for refresh rate
                                <1> vbe3_sm_4:
000018D6 80E7C1                 <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
                                <1>       ;mov   [vbe_mode_x], bh
                                <1>
000018D9 803D[9A6E0000]03       <1>       cmp    byte [CRT_MODE], 3 ; is current mode 03h ?
000018E0 7509                   <1>       jne    short vbe3_sm_1    ; no
                                <1>
                                <1>       ; save mode 03h video pages and cursor positions
000018E2 57                     <1>       push   edi ; **!***
000018E3 51                     <1>       push   ecx ; ******
                                <1>       ;push  esi
                                <1>
000018E4 E8CC040000             <1>       call   save_mode3_multiscreen
                                <1>
                                <1>       ;pop   esi
000018E9 59                     <1>       pop    ecx ; ******
000018EA 5F                     <1>       pop    edi ; **!***
                                <1> vbe3_sm_1:
                                <1>       ; ax = 4F02h
                                <1>       ; bx = video mode number (vbe2 type)
000018EB E8B4000000             <1>       call   int10h_32bit_pmi
                                <1>                     ; call to far call to VBE3 PMI
                                <1>
000018F0 6683F84F               <1>       cmp    ax, 004Fh ; succeeded ?
000018F4 750E                   <1>       jne    short vbe3_sm_2
                                <1>       ; set current mode byte/sign to extended (SVGA) mode
000018F6 C605[9A6E0000]FF       <1>       mov    byte [CRT_MODE], 0FFh ; VESA VBE mode
                                <1>       ; set current VBE mode word to bx input
000018FD 66891D[1E120300]       <1>       mov    [video_mode], bx
                                <1> vbe3_sm_2:
                                <1>       ; 22/12/2020
00001904 5F                     <1>       pop    edi ; *****
00001905 C3                     <1>       retn
                                <1>
                                <1> vbe3_sm_3:
                                <1>       ; 22/12/2020
                                <1>       ; copy user's CRTCInfoBlock to the buffer
00001906 51                     <1>       push   ecx
00001907 89FE                   <1>       mov    esi, edi
00001909 BF807D0900             <1>       mov    edi, VBE3CRTCINFOBLOCK
0000190E B940000000             <1>       mov    ecx, 64
00001913 E856010100             <1>       call   transfer_from_user_buffer
00001918 59                     <1>       pop    ecx
                                <1>       ; set offset (es base addr is VBE3SAVERESTOREBLOCK)
00001919 81EF00760900           <1>       sub    edi, VBE3SAVERESTOREBLOCK
0000191F EBB5                   <1>       jmp    short vbe3_sm_4
                                <1>
                                <1> vesa_vbe3_pmi:
                                <1>       ; 12/12/2020
                                <1>       ; 08/12/2020
                                <1>       ; 07/12/2020
                                <1>       ; 05/12/2020, 06/12/2020
                                <1>       ; 03/12/2020, 04/12/2020
                                <1>       ; 28/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; VGA BIOS functions via
                                <1>       ; VESA VBE3 Protected Mode Inface
                                <1>       ; [vbe3] = 3 and [pmi32] > 0
                                <1>
                                <1>       ; 04/12/2020
                                <1>       ; Only 'set mode' will be redirected to vbe3 video bios
                                <1>       ; (by setting mode 3 multiscreen paraters before and after)
                                <1>
                                <1>       ; 06/12/2020
00001921 20E4                   <1>       and    ah, ah ; 0 = set mode function
00001923 7402                   <1>       jz     short vbe3_pmi_0
00001925 EB76                   <1>       jmp    vbe3_pmi_9
```

```
                                <1>
                                <1> vbe3_pmi_0:
                                <1>       ; 07/12/2020
00001927 88C4                   <1>       mov   ah, al
00001929 80E480                 <1>       and   ah, 80h ; 0 or 80h
0000192C 30E0                   <1>       xor   al, ah ; 8?h -> 0?h
                                <1>
                                <1>       ;cmp   al, 13h ; mode number above 13h is returned
                                <1>       ;jna   short vbe3_pmi_1
                                <1>       ;            ; back to default code due to uncertainty
                                <1>       ;            ; (>13h is not std for all svga bioses)
                                <1>       ;jmp   VGA_funcs_0
                                <1> vbe3_pmi_1:
                                <1>       ; 07/12/2020
                                <1>       ; Possible cases for VBE3 (PMI, ah=0) set mode:
                                <1>       ; current mode > 07h and requested mode: any
                                <1>       ; current mode <= 07h and requested mode > 07h
                                <1>
                                <1>       ; 06/12/2020
0000192E 8825[87950100]         <1>       mov   byte [noclearmem], ah ; 0 or 80h
                                <1>       ; check current video mode if it is 03h
00001934 803D[9A6E0000]03       <1>       cmp   byte [CRT_MODE], 3 ; current mode
0000193B 750B                   <1>       jne   short vbe3_pmi_3
                                <1>       ; 07/12/2020
                                <1>       ; check new video mode if it is 03h also
                                <1>       ;cmp   al, 3
                                <1>       ;jne   short vbe3_pmi_2
                                <1>       ;mov   byte [p_crt_mode], 80h    ; clear video memory
                                <1>       ;jmp   short vbe3_pmi_5
                                <1> vbe3_pmi_2:
                                <1>       ; case 1:
                                <1>       ; Current mode is 03h and new mode is not 03h
                                <1>
                                <1>       ; save video pages and cursor positions
0000193D 56                     <1>       push  esi
0000193E 57                     <1>       push  edi
0000193F 51                     <1>       push  ecx
                                <1>
                                <1>       ; 12/12/2020
                                <1>       ;mov   esi, 0B8000h ; mode 3 video memory
                                <1>       ;mov   edi, 98000h  ; backup location
                                <1>       ;mov   ecx, (0B8000h-0B0000h)/4
                                <1>       ;rep   movsd
                                <1>       ;
                                <1>       ;mov   byte [p_crt_mode], 3 ; previous mode, backup sign
                                <1>       ;xchg  cl, [ACTIVE_PAGE]
                                <1>       ;mov   [p_crt_page], cl  ; save as previous active page
                                <1>       ;
                                <1>       ;; save cursor positions
                                <1>       ;mov   esi, CURSOR_POSN
                                <1>       ;mov   edi, cursor_pposn ; cursor positions backup
                                <1>       ;mov   cl, 4
                                <1>       ;rep   movsd
                                <1>
                                <1>       ; 12/12/2020
00001940 E870040000             <1>       call  save_mode3_multiscreen
                                <1>
00001945 59                     <1>       pop   ecx
00001946 5F                     <1>       pop   edi
00001947 5E                     <1>       pop   esi
                                <1> vbe3_pmi_3:
                                <1>       ; 08/12/2020
                                <1>       ; 07/12/2020
                                <1>       ; case 3 or case 4
00001948 A2[9A6E0000]           <1>       mov   [CRT_MODE], al
0000194D 3C03                   <1>       cmp   al, 3
0000194F 7407                   <1>       je    short vbe3_pmi_4
                                <1>       ; case 4:
                                <1>       ; Current mode is not 03h and also new mode is not 03h
00001951 800D[85950100]80       <1>       or    byte [p_crt_mode], 80h  ; 83h (case 1 -> case 4)
                                <1>       ;jmp   short vbe3_pmi_5
                                <1> vbe3_pmi_4:
                                <1>       ; case 3:
                                <1>       ;
                                <1>       ; Current mode is not 03h and new mode is 03h
                                <1>
                                <1> ;vbe3_pmi_5:
                                <1>       ;mov   [CRT_MODE], al
                                <1>
00001958 E847000000             <1>       call  int10h_32bit_pmi
                                <1>
0000195D 803D[9A6E0000]03       <1>       cmp   byte [CRT_MODE], 3  ; new video mode
                                <1>       ;jne   vbe3_pmi_8   ; video mode <> 03h
00001964 7532                   <1>       jne   short vbe3_pmi_8
                                <1>
                                <1>       ;push  eax ; 04/12/2020
00001966 53                     <1>       push  ebx
00001967 51                     <1>       push  ecx
00001968 52                     <1>       push  edx
00001969 57                     <1>       push  edi ; 03/12/2020
                                <1>
                                <1>       ; 12/12/2020
0000196A 56                     <1>       push  esi
0000196B E878040000             <1>       call  restore_mode3_multiscreen
00001970 5E                     <1>       pop   esi
                                <1>       ; AL = active video page
                                <1>
                                <1>       ; 12/12/2020
                                <1>       ;mov   al, [p_crt_page] ; previous mode 3 active page
                                <1>       ;
                                <1>       ;;test byte [p_crt_mode], 7Fh ; 83h or 80h or 03h
                                <1>       ;;jz   short vbe3_pmi_6 ; do not restore video pages
                                <1>       ;                   ; clear current video page
                                <1>       ;; case 3
                                <1>       ;
```

```
3631                                 <1>       ;; ([p_crt_mode] = 03h)
3632                                 <1>       ;
3633                                 <1>       ;; New video mode is 3 while current video mode is not 3
3634                                 <1>       ;; (multi screen) video pages will be restored from 098000h
3635                                 <1>       ;
3636                                 <1>       ;; restore video pages and cursor positions
3637                                 <1>       ;
3638                                 <1>       ;mov   [ACTIVE_PAGE], al ; current mode 3 active page
3639                                 <1>       ;
3640                                 <1>       ;push  esi
3641                                 <1>       ;
3642                                 <1>       ;; restore video pages
3643                                 <1>       ;mov   esi, 98000h
3644                                 <1>       ;mov   edi, 0B8000h
3645                                 <1>       ;;mov  ecx, 2000h
3646                                 <1>       ;mov   cx, 2000h ; 8K dwords (32K)
3647                                 <1>       ;rep   movsd
3648                                 <1>       ;
3649                                 <1>       ;mov   [p_crt_mode], cl ; reset ('case 3' end condition)
3650                                 <1>       ;
3651                                 <1>       ;; restore cursor positions
3652                                 <1>       ;mov   esi, cursor_pposn
3653                                 <1>       ;mov   edi, CURSOR_POSN
3654                                 <1>       ;;mov  ecx, 4 ; restore all cursor positions (16 bytes)
3655                                 <1>       ;mov   cl, 4
3656                                 <1>       ;rep   movsd
3657                                 <1>       ;
3658                                 <1>       ;pop   esi
3659                                 <1>       ;
3660                                 <1>       ;; 07/12/2020
3661                                 <1>       ;; restore CRT_START according to ACTIVE_PAGE
3662                                 <1>       ;mov   [CRT_START], cx ; 0
3663                                 <1>       ;
3664                                 <1>       ;; check active page and set it again if it is not 0
3665                                 <1>       ;or    al, al
3666                                 <1>       ;jz    short vbe3_pmi_7
3667                                 <1>       ;
3668                                 <1>       ;mov   cl, al
3669                                 <1> ;vbe3_pmi_5:
3670                                 <1>       ;add   word [CRT_START], 4096
3671                                 <1>       ;dec   cl
3672                                 <1>       ;jnz   short vbe3_pmi_5
3673                                 <1>
3674 00001971 B405                  <1>       mov    ah, 05h ; set current video page
3675                                 <1>       ;al = video page
3676 00001973 E82C000000            <1>       call   int10h_32bit_pmi
3677                                 <1>
3678                                 <1>       ; check current cursor position & set it again if not 0,0
3679                                 <1>       ;movzx ebx, byte [ACTIVE_PAGE]
3680 00001978 0FB6D8                <1>       movzx  ebx, al
3681 0000197B D0E3                  <1>       shl    bl, 1
3682 0000197D 81C3[0E890100]        <1>       add    ebx, CURSOR_POSN
3683 00001983 668B13                <1>       mov    dx, [ebx]
3684 00001986 6621D2                <1>       and    dx, dx
3685 00001989 7409                  <1>       jz     short vbe3_pmi_7
3686                                 <1>
3687                                 <1>       ;dx = cursor position (dl = column, dh = row)
3688                                 <1>       ;mov   bh, [ACTIVE_PAGE] ; 06/12/2020
3689 0000198B 88C7                  <1>       mov    bh, al
3690 0000198D B402                  <1>       mov    ah, 02h ; set cursor position
3691 0000198F E810000000            <1>       call   int10h_32bit_pmi
3692                                 <1>
3693                                 <1>       ;jmp   short vbe3_pmi_7
3694                                 <1>
3695                                 <1> ;vbe3_pmi_6:
3696                                 <1> ;     ; 07/12/2020
3697                                 <1> ;     ; case 1, previous mode is 03h, current mode is 03h
3698                                 <1> ;     ; 03/12/2020
3699                                 <1> ;     cmp    byte [noclearmem], 0
3700                                 <1> ;     jna    short vbe3_pmi_7 ; do not clear memory
3701                                 <1> ;     ; clear video page
3702                                 <1> ;     mov    ecx, 1024 ; 4096/4
3703                                 <1> ;     mov    eax, 07200720h
3704                                 <1> ;     mov    edi, 0B8000h ; [crt_base]
3705                                 <1> ;     add    di, [CRT_START]
3706                                 <1> ;     rep    stosd  ; FILL THE REGEN BUFFER WITH BLANKS
3707                                 <1>
3708                                 <1> vbe3_pmi_7:
3709 00001994 5F                    <1>       pop    edi
3710 00001995 5A                    <1>       pop    edx
3711 00001996 59                    <1>       pop    ecx
3712 00001997 5B                    <1>       pop    ebx
3713                                 <1>       ;pop   eax ; 04/12/2020
3714                                 <1> vbe3_pmi_8:
3715                                 <1>       ; 04/12/2020
3716                                 <1>       ;(TRDOS 386 v2.0.3, INT 31h, ah=0 return)
3717 00001998 31C0                  <1>       xor    eax, eax  ; eax = 0 -> succesful
3718                                 <1> vesa_vbe3_pmi_retn:
3719 0000199A 07                    <1>       pop    es  ; **
3720 0000199B 1F                    <1>       pop    ds  ; *
3721 0000199C CF                    <1>       iretd
3722                                 <1>
3723                                 <1> vbe3_pmi_9:
3724                                 <1>       ; 06/12/2020
3725                                 <1>       ;cmp   ah, 10h ; Set/Get Palette Registers
3726                                 <1>       ;jnb   short vbe3_pmi_10
3727                                 <1>       ; 05/12/2020
3728 0000199D E802000000            <1>       call   int10h_32bit_pmi
3729 000019A2 EBF6                  <1>       jmp    short vesa_vbe3_pmi_retn
3730                                 <1>
3731                                 <1> ;vbe3_pmi_10:
3732                                 <1>       ; 06/12/2020
3733                                 <1>       ;jmp   VGA_funcs_0
3734                                 <1>
3735                                 <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
000019A4 56             <1>       push   esi
000019A5 C1E010         <1>       shl    eax, 16  ; move function number (ax) to hw
000019A8 8B35[24120300] <1>       mov    esi, [pmid_addr] ; linear address of
                        <1>                               ; PMInfo.Entrypoint pointer
                        <1>       ;mov   ax, [esi+PMInfo.EntryPoint]
000019AE 668B06         <1>       mov    ax, [esi]
000019B1 C1C010         <1>       rol    eax, 16 ; move PM entry address to hw
                        <1>                      ; and move function number to lw (ax)
000019B4 5E             <1>       pop    esi
                        <1>
                        <1>       ; top of stack: ; (*)
                        <1>       ; return (the caller) address of "int10h_32bit_pmi"
                        <1>
000019B5 E9E0EDFFFF     <1>       jmp    _VBE3PMI_fcall ; will return to the caller (*)
                        <1>
                        <1> _vbe3_pmfn_srs_8:
                        <1>       ; 17/01/2021
000019BA 31DB           <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
                        <1> _vbe3_pmfn_srs_9: ; 24/01/2021
000019BC 66B8044F       <1>       mov    ax, 4F04h
000019C0 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>
000019C2 21DB           <1>       and    ebx, ebx ; user's buffer address
000019C4 750A           <1>       jnz    short _vbe3_pmfn_save_restore_state
                        <1>
000019C6 08D2           <1>       or     dl, dl
000019C8 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:
000019CA B84F010000     <1>       mov    eax, 014Fh ; ah = 1 : Function call failed
                        <1>                         ; al = 4Fh : Function is supported
                        <1> _vbe3_srs_retn:
000019CF C3             <1>       retn
                        <1>
                        <1> _vbe3_pmfn_save_restore_state:
000019D0 20D2           <1>       and    dl, dl
000019D2 7559           <1>       jnz    short _vbe3_pmfn_srs_2
                        <1> _vbe3_pmfn_srs:
000019D4 31DB           <1>       xor    ebx, ebx
                        <1> _vbe3_pmfn_srs_0:
                        <1>       ; 24/01/2021
000019D6 83F90F         <1>       cmp    ecx, 0Fh
                        <1>       ;ja    short _vbe3_pmfn_srs_1
000019D9 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)
000019DB 80E1FD         <1>       and    cl, ~2 ; and cl, not 2
                        <1>
                        <1>       ; 24/01/2021
                        <1>       ;mov   bl, 1
000019DE FEC3           <1>       inc    bl ; = 1
000019E0 66D3E3         <1>       shl    bx, cl
000019E3 66231D[28120300] <1>     and    bx, [vbe3stbsflags]
000019EA 7416           <1>       jz     short _vbe3_pmfn_srs_1
                        <1>       ;mov   bx, cx
000019EC 89CB           <1>       mov    ebx, ecx ; <= 15
000019EE D0E3           <1>       shl    bl, 1 ; 0, 2, 8 .. 30
000019F0 668B9B[DE3B0000] <1>     mov    bx, [vbestatebufsize+ebx]
000019F7 89DF           <1>       mov    edi, ebx
                        <1>       ; edi = state buffer size in bytes
000019F9 66C1EB06       <1>       shr    bx, 6 ; / 64
000019FD 66B84F00       <1>       mov    ax, 4Fh
00001A01 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
00001A02 E8B5FFFFFF             <1>       call   _vbe3_pmfn_srs_9
00001A07 6683F84F               <1>       cmp    ax, 004Fh
00001A0B 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
00001A0D B001                   <1>       mov    al, 1
00001A0F 66D3E0                 <1>       shl    ax, cl
00001A12 660905[28120300]       <1>       or     [vbe3stbsflags], ax ; set flag for state option
                                <1>       ; 23/01/2021
00001A19 89DF                   <1>       mov    edi, ebx
00001A1B 89C8                   <1>       mov    eax, ecx
00001A1D D0E0                   <1>       shl    al, 1
00001A1F 66C1E706               <1>       shl    di, 6 ; * 64
00001A23 6689B8[DE3B0000]       <1>       mov    [vbestatebufsize+eax], di
                                <1>                              ; save buf size for option
                                <1>       ;xchg  edi, ebx
                                <1>       ; edi = state buffer size in bytes
00001A2A B04F                   <1>       mov    al, 4Fh
00001A2C 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>
00001A2D F6C1FD                 <1>       test   cl, ~2 ; test cl, not 2
00001A30 7498                   <1>       jz     short _vbe3_pmfn_srs_fail
                                <1>
00001A32 80FA02                 <1>       cmp    dl, 2
00001A35 7748                   <1>       ja     short _vbe3_pmfn_srs_5
                                <1>
                                <1>       ;and   cl, ~2 ; and cl, not 2
                                <1>
00001A37 53                     <1>       push   ebx ; * ; buffer address
                                <1>       ; save or restore state
                                <1>       ; (get required buffer size at first)
00001A38 52                     <1>       push   edx ; **
00001A39 28D2                   <1>       sub    dl, dl ; 0
00001A3B E894FFFFFF             <1>       call   _vbe3_pmfn_srs
00001A40 5A                     <1>       pop    edx ; **
                                <1>       ; 24/01/2021
00001A41 5B                     <1>       pop    ebx ; *
00001A42 08E4                   <1>       or     ah, ah
00001A44 7538                   <1>       jnz    short _vbe3_pmfn_srs_4 ; error
                                <1>
                                <1>       ; edi = buffer size in bytes
00001A46 81FF00080000           <1>       cmp    edi, 2048
00001A4C 772B                   <1>       ja     short _vbe3_pmfn_srs_3
                                <1>
00001A4E 80FA01                 <1>       cmp    dl, 1
00001A51 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
00001A53 E842000000             <1>       call   _vbe3_pmfn_srs_7
                                <1>
00001A58 6683F84F               <1>       cmp    ax, 004Fh
00001A5C 7520                   <1>       jne    short _vbe3_pmfn_srs_4
                                <1>
00001A5E 09DB                   <1>       or     ebx, ebx  ; kernel ('sysvideo') ?
00001A60 741C                   <1>       jz     short _vbe3_pmfn_srs_4 ; yes
                                <1>
                                <1>       ; the caller is user
00001A62 51                     <1>       push   ecx ; *
00001A63 89F9                   <1>       mov    ecx, edi ; state buffer size
00001A65 BE00760900             <1>       mov    esi, VBE3SAVERESTOREBLOCK ; source
                                <1>                                ; (vbe3 pmi buff)
00001A6A 89DF                   <1>       mov    edi, ebx     ; destination (user buff)
00001A6C E8B3FF0000             <1>       call   transfer_to_user_buffer
00001A71 59                     <1>       pop    ecx ; *
00001A72 7205                   <1>       jc     short _vbe3_pmfn_srs_3
                                <1>
00001A74 29C0                   <1>       sub    eax, eax
00001A76 B04F                   <1>       mov    al, 4Fh
00001A78 C3                     <1>       retn
                                <1>
                                <1>       ; 24/01/2021
                                <1> _vbe3_pmfn_srs_3:
00001A79 B84F010000             <1>       mov    eax, 014Fh
                                <1> _vbe3_pmfn_srs_4:
00001A7E C3                     <1>       retn
                                <1> _vbe3_pmfn_srs_5:
00001A7F 31C0                   <1>       xor    eax, eax
00001A81 FEC4                   <1>       inc    ah
                                <1>       ; eax = 0100h, function is not supported
00001A83 C3                     <1>       retn
                                <1>
                                <1> _vbe3_pmfn_srs_6:
                                <1>       ; restore video state
                                <1>       ; 24/01/2021
                                <1>       ;pop   ebx ; *
```

```
                                <1>       ; 23/01/2021
00001A84 09DB                   <1>       or     ebx, ebx ; 0 ?
00001A86 7412                   <1>       jz     short _vbe3_pmfn_srs_7 ; 'sysvideo' call
                                <1>       ; 24/01/2021
                                <1>       ;jz    _vbe3_pmfn_srs_8
00001A88 89DE                   <1>       mov    esi, ebx
                                <1>       ; esi = user's video state buffer
00001A8A 51                     <1>       push   ecx ; *
00001A8B 89F9                   <1>       mov    ecx, edi ; state buffer size
00001A8D BF00760900             <1>       mov    edi, VBE3SAVERESTOREBLOCK ; destination
                                <1>                                 ; (vbe3 pmi buff)
                                <1>       ;mov   esi, ebx     ; source (user buff)
00001A92 E8D7FF0000             <1>       call   transfer_from_user_buffer
00001A97 59                     <1>       pop    ecx ; *
00001A98 72DF                   <1>       jc     short _vbe3_pmfn_srs_3
                                <1> _vbe3_pmfn_srs_7:
00001A9A 53                     <1>       push   ebx ; *
                                <1>       ; restore video state
                                <1>       ;xor   ebx, ebx ; points to VBE3SAVERESTOREBLOCK
                                <1>       ;mov   ax, 4F04h
                                <1>       ;call  int10h_32bit_pmi
                                <1>       ; 17/01/2021
00001A9B E81AFFFFFF             <1>       call   _vbe3_pmfn_srs_8
00001AA0 5B                     <1>       pop    ebx ; *
00001AA1 C3                     <1>       retn
                                <1>
                                <1> VIDEO_STATE:
                                <1>       ; 26/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> ;--------------------------------------------------
                                <1> ; VIDEO STATE
                                <1> ;  RETURNS THE CURRENT VIDEO STATE IN AX
                                <1> ;  AH = NUMBER OF COLUMNS ON THE SCREEN
                                <1> ;  AL = CURRENT VIDEO MODE
                                <1> ;  BH = CURRENT ACTIVE PAGE
                                <1> ;--------------------------------------------------
                                <1>
00001AA2 8A25[9C6E0000]         <1>       mov    ah, [CRT_COLS]      ; GET NUMBER OF COLUMNS
00001AA8 A0[9A6E0000]           <1>       mov    al, [CRT_MODE]      ; CURRENT MODE
                                <1>       ;movzx esi, al
                                <1>       ;mov   ah, [esi+M6]
                                <1>       ; BH = active page
00001AAD 8A3D[1E890100]         <1>       mov    bh, [ACTIVE_PAGE] ; GET CURRENT ACTIVE PAGE
00001AB3 FA                     <1>       cli    ; 02/01/2017
00001AB4 5D                     <1>       pop    ebp          ; RECOVER REGISTERS
00001AB5 5F                     <1>       pop    edi
00001AB6 5E                     <1>       pop    esi
00001AB7 59                     <1>       pop    ecx          ; DISCARD SAVED BX
00001AB8 EB41                   <1>       jmp    short M15    ; RETURN TO CALLER
                                <1>
                                <1> set_mode_ncm:
                                <1>       ; 17/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; 04/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; set mode without clearing the video memory
                                <1>       ; (ony for graphics modes)
                                <1>
                                <1>       ;cmp   al, 7 ; IBM PC CGA modes
                                <1>       ;jna   short SET_MODE ; normal function (clear)
                                <1>       ;; do not clear memory
                                <1>       ;;mov  [noclearmem], al ; > 0
                                <1>       ;mov   byte [noclearmem], 80h ; 17/11/2020
                                <1>       ;call  _set_mode
                                <1>       ;mov   byte [noclearmem], 0
                                <1>         ;jmp     short VIDEO_RETURN
                                <1>
                                <1>       ; 17/11/2020 (TRDOS v2.0.3)
00001ABA 0C80                   <1>       or     al, 80h ; not clear memory option
                                <1>
                                <1>       ; 05/12/2020
                                <1>       ; 27/11/2020
                                <1>       ; 17/11/2020
                                <1>       ; 08/08/2016, 10/08/2016
                                <1>       ; 29/07/2016, 30/07/2016
                                <1>       ; 25/07/2016, 26/07/2016, 27/07/2016
                                <1>       ; 02/07/2016, 18/07/2016, 23/07/2016
                                <1>       ; 24/06/2016, 26/06/2016
                                <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> SET_MODE:
                                <1>       ; For 32 bit TRDOS and Retro UNIX 386:
                                <1>       ;     valid video mode: 03h only!
                                <1>       ;     (VGA modes will be selected with another routine)
                                <1>       ;
                                <1>       ; set_txt_mode ; 80*25 (16 fore colors, 8 back colors)
                                <1>
                                <1>       ; 27/11/2020
                                <1>
                                <1>       ; Check if current mode is
                                <1>       ; Bochs/Plex86 VBE graphics mode
00001ABC 803D[9A6E0000]FF       <1>       cmp    byte [CRT_MODE], 0FFh ; VESA VBE graphics mode
00001AC3 7220                   <1>       jb     short _set_mode_      ; signature
                                <1>                                    ; VBE mode number is in
                                <1>                                    ; [video_mode] bit 0to8
00001AC5 88C3                   <1>       mov    bl, al ; save video mode
00001AC7 E866240000             <1>       call   dispi_get_enable
00001ACC 50                     <1>       push   eax ; save current VBE dispi status
                                <1>       ; Disable Bochs/Plex86 VBE dispi
                                <1>       ;mov   ax, 0 ; VBE_DISPI_DISABLED
00001ACD 31C0                   <1>       xor    eax, eax ; 0
00001ACF E832240000             <1>       call   dispi_set_enable
00001AD4 88D8                   <1>       mov    al, bl ; restore video mode
00001AD6 E827000000             <1>       call   _set_mode
00001ADB 58                     <1>       pop    eax ; restore current VBE dispi status
00001ADC 7313                   <1>       jnc    short VIDEO_RETURN
```

```
                                <1>       ; ! unimplemented or invalid video mode number !
                                <1>       ; VBE dispi must be enabled again
                                <1>       ; (return to run on current VBE graphics mode)
                                <1>       ;;mov  al, [video_mode+1] ; bit 8 to 15
                                <1>       ;;and  al, 0C0h ; isolate bit 14 and bit 15
                                <1>       ;;or   al, 1 ; VBE_DISPI_ENABLED
00001ADE E823240000             <1>       call   dispi_set_enable
00001AE3 EB07                   <1>       jmp    short _video_func_err
                                <1>
                                <1> _set_mode_:
                                <1>       ; VGA bios (non-VBE) 'setmode' procedure
                                <1>
                                <1>       ; 26/11/2020 (TRDOS v2.0.3)
                                <1>
                                <1> ;--------------------------------------------------
                                <1> ; SET MODE                                         :
                                <1> ;     THIS ROUTINE INITIALIZES THE ATTACHMENT TO    :
                                <1> ;     THE SELECTED MODE, THE SCREEN IS BLANKED.     :
                                <1> ; INPUT                                            :
                                <1> ;     (AL) - MODE SELECTED (RANGE 0-7)             :
                                <1> ; OUTPUT                                           :
                                <1> ;     NONE                                         :
                                <1> ;--------------------------------------------------
                                <1>
00001AE5 E818000000             <1>       call   _set_mode ; 24/06/2016 (set_txt_mode)
                                <1>       ; 26/11/2020
00001AEA 7305                   <1>       jnc    short VIDEO_RETURN
                                <1>
                                <1>       ; 26/11/2020
                                <1> _video_func_err:
00001AEC 31C0                   <1>       xor    eax, eax ; function call failed
00001AEE 48                     <1>       dec    eax  ; 0FFFFFFFFh ; - 1
00001AEF EB05                   <1>       jmp    short _video_return
                                <1>
                                <1> ; 12/05/2016
                                <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
                                <1>
                                <1> VIDEO_RETURN:
00001AF1 A1[78950100]           <1>       mov    eax, [video_eax] ; 12/05/2016
                                <1> _video_return:
00001AF6 FA                     <1>       cli ; 02/01/2017
00001AF7 5D                     <1>       pop    ebp ; ******** ; 26/11/2020
00001AF8 5F                     <1>       pop    edi ; *******
00001AF9 5E                     <1>       pop    esi ; ******
00001AFA 5B                     <1>       pop    ebx ; *****
                                <1> M15:  ; VIDEO_RETURN_C
                                <1>       ;;15/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [intflg], 0
                                <1>       ;
00001AFB 59                     <1>       pop    ecx ; **** ; 26/11/2020
00001AFC 5A                     <1>       pop    edx ; ***
00001AFD 1F                     <1>       pop    ds  ; **
00001AFE 07                     <1>       pop    es  ; * ; RECOVER SEGMENTS
00001AFF CF                     <1>       iretd        ; ALL DONE
                                <1>
                                <1> set_txt_mode:
                                <1>
                                <1>       ; 29/07/2016
                                <1>       ; 27/06/2016
00001B00 B003                   <1>       mov    al, 3 ; 26/11/2020 (bit 7 = 0)
                                <1>
                                <1>       ; 17/11/2020 (TRDOS v2.0.3)
                                <1>       ;mov  byte [noclearmem], 0
                                <1>
                                <1> ; 12/04/2021
                                <1> ; 10/08/2016
                                <1> ; 08/08/2016
                                <1> ; 30/07/2016
                                <1> ; 29/07/2016
                                <1> ; 25/07/2016, 26/07/2016, 27/07/2016
                                <1> ; 07/07/2016, 18/07/2016, 23/07/2016
                                <1> ; 02/07/2016, 03/07/2016, 04/07/2016
                                <1> ; 26/06/2016
                                <1> ; 24/06/2016 (set_txt_mode -> _set_mode)
                                <1> ; 17/06/2016
                                <1> ; 29/05/2016
                                <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> _set_mode:
                                <1>       ; 12/12/2020
                                <1>       ; 26/11/2020
                                <1>       ; call from 'biosfn_set_video_mode'
                                <1>       ;      (bochs/plex86 video bios code)
                                <1>       ; call from 'SET_MODE'
                                <1>       ;      (TRDOS 386 v2 default, IBM PC/AT rom bios code)
                                <1>       ; continue from 'set_txt_mode'
                                <1>
                                <1>       ; INPUT:
                                <1>       ;      al = VGA video mode
                                <1>       ; RETURN:
                                <1>       ;      cf = 1 -> video mode not implemented
                                <1>       ;      cf = 0 -> OK
                                <1>       ;
                                <1>       ; Modified registers: eax, bx, ecx, esi, edi, (ebp)
                                <1>
                                <1>       ; 17/11/2020 (TRDOS v2.0.3)
                                <1>       ; no clear memory option
                                <1>       ; (from mode number byte bit 7)
00001B02 88C4                   <1>       mov    ah, al
00001B04 80E480                 <1>       and    ah, 80h
                                <1>       ;mov   [noclearmem], al
00001B07 8825[87950100]         <1>       mov    [noclearmem], ah
```

```
4156                                    <1>     ;and   al, 7Fh ; clear bit 7
4157                                    <1>     ;;xor  [noclearmem], al ; clear bit 0 to 6
4158                                    <1>     ; 26/11/2020
4159 00001B0D 30E0                      <1>     xor    al, ah ; and al, 7Fh
4160                                    <1>
4161                                    <1>     ; 19/11/2020
4162                                    <1>
4163                                    <1>     ; Video mode 03h action principle:
4164                                    <1>     ;
4165                                    <1>     ; for case 1:
4166                                    <1>     ; Current mode is 03h and next/requested mode is not 03h
4167                                    <1>     ;      - save mode (set mode 03h flag)
4168                                    <1>     ;      - save 8 video pages (which are will be restored)
4169                                    <1>     ;      - save active page number (which will be reactivated)
4170                                    <1>     ;      - set active page to 0 always (no multi screen)
4171                                    <1>     ;      - save 8 cursor positions (which will be restored)
4172                                    <1>     ;      - use 'noclearmem' option
4173                                    <1>     ;      [p_crt_mode] = 0 -> 03h
4174                                    <1>     ;
4175                                    <1>     ; for case 2:
4176                                    <1>     ; Current mode is 03h and next/requested mode is also 03h
4177                                    <1>     ;      - clear active video page if 'noclearmem' is not set
4178                                    <1>     ;      [p_crt_mode] = 0 -> 80h -> 0
4179                                    <1>     ;
4180                                    <1>     ; for case 3:
4181                                    <1>     ; Current mode is not 03h and next/requested mode is 03h
4182                                    <1>     ;      - restore video pages (8 video pages were saved)
4183                                    <1>     ;      - restore active page number (which were saved)
4184                                    <1>     ;      - restore 8 cursor positions (which were saved)
4185                                    <1>     ;      - reset/clear mode 03h flag
4186                                    <1>     ;      [p_crt_mode] = 03h -> 0
4187                                    <1>     ;
4188                                    <1>     ; for case 4:
4189                                    <1>     ; Current mode is not 03h and next/requested mode is not 03h
4190                                    <1>     ;      - use 'noclearmem' option
4191                                    <1>     ;      - set active page to 0 always
4192                                    <1>     ;      [p_crt_mode] = 03h -> 83h -> 03h
4193                                    <1>     ;
4194                                    <1>     ; initial (boot time) values:
4195                                    <1>     ;      [p_crt_mode] = 0 ("there isn't a page backup, yet")
4196                                    <1>     ;        [CRT_MODE] = 3 (kernel's starting mode)
4197                                    <1>
4198                                    <1>     ; 26/11/2020
4199 00001B0F 3C03                      <1>     cmp    al, 03h      ; mode 3, 80x25 text, 16 colors
4200 00001B11 7515                      <1>     jne    short _sm_0 ; (default mode for TRDOS 386 mainprog)
4201                                    <1>
4202                                    <1>     ; case 2 or case 3
4203                                    <1>
4204                                    <1>     ; check current video mode if it is 03h
4205 00001B13 08E4                      <1>     or     ah, ah ; 80h or 0 ('noclearmem' option)
4206 00001B15 7521                      <1>     jnz    short _sm_1 ; do not clear display page
4207                                    <1>
4208                                    <1>     ; 26/11/2020
4209                                    <1>     ; Note:
4210                                    <1>     ; [CRT_MODE] = 0FFh for VESA VBE video modes
4211                                    <1>     ; [video_mode] = standard VGA and VESA VBE video modes
4212                                    <1>
4213 00001B17 3805[9A6E0000]            <1>     cmp    [CRT_MODE], al      ; 03h
4214 00001B1D 7520                      <1>     jne    short _sm_2  ; case 3 ([p_crt_mode] = 03h)
4215                                    <1>
4216                                    <1>     ; case 2
4217                                    <1>
4218                                    <1>     ; [p_crt_mode] = 0
4219                                    <1>
4220                                    <1>     ; 19/11/2020
4221                                    <1>     ; If '_set_mode' procedure is called for video mode 3
4222                                    <1>     ;     while video mode is 3, video page will be cleared
4223                                    <1>     ;     and cursor position of video page will be reset.
4224                                    <1>
4225                                    <1>     ; clear display page
4226 00001B1F C605[85950100]80          <1>     mov    byte [p_crt_mode], 80h ; clear page sign
4227 00001B26 EB1C                      <1>     jmp    short _sm_3  ; bypass save video page routine
4228                                    <1> _sm_0:
4229                                    <1>     ; case 1 or case 4
4230                                    <1>
4231                                    <1>     ; 05/12/2020
4232 00001B28 803D[9A6E0000]03          <1>     cmp    byte [CRT_MODE], 3 ; is current mode 03h?
4233 00001B2F 7507                      <1>     jne    short _sm_1  ; case 4 ; [p_crt_mode] = 03h
4234                                    <1>
4235                                    <1>     ; case 1
4236                                    <1>     ; [p_crt_mode] = 0
4237                                    <1>
4238                                    <1>     ; 19/11/2020
4239                                    <1>     ; If '_set_mode' procedure is called for a video mode
4240                                    <1>     ;     except video mode 3 while current video mode
4241                                    <1>     ;     is 3, all video pages of mode 3 will be copied
4242                                    <1>     ;     to 98000h address as backup, before mode change.
4243                                    <1>
4244                                    <1> _sm_save_pm:
4245                                    <1>     ; 12/12/2020
4246                                    <1>     ;; 03/07/2016
4247                                    <1>     ;; save video pages
4248                                    <1>     ;mov   esi, 0B8000h
4249                                    <1>     ;mov   edi, 98000h ; 30/07/2016
4250                                    <1>     ;mov   ecx, (0B8000h-0B0000h)/4
4251                                    <1>     ;rep   movsd
4252                                    <1>
4253                                    <1>     ;mov   byte [p_crt_mode], 3 ; previous mode, backup sign
4254                                    <1>     ;; 26/11/2020
4255                                    <1>     ;xchg  cl, [ACTIVE_PAGE]
4256                                    <1>     ;mov   [p_crt_page], cl  ; save as previous active page
4257                                    <1>     ;
4258                                    <1>     ;; save cursor positions
4259                                    <1>     ;mov   esi, CURSOR_POSN
4260                                    <1>     ;mov   edi, cursor_pposn ; cursor positions backup
```

```
                                <1>       ;mov   cl, 4
                                <1>       ;rep   movsd
                                <1>
                                <1>       ; 12/12/2020
00001B31 E87F020000             <1>       call   save_mode3_multiscreen
                                <1>
                                <1>       ; 29/07/2016
                                <1>       ;;mov  [ACTIVE_PAGE], cl ; 0
                                <1>       ;xchg  cl, [ACTIVE_PAGE]
                                <1>       ;mov   [p_crt_page], cl  ; previous page (for mode 3)
                                <1>
                                <1>       ; [ACTIVE_PAGE] = 0
                                <1>
00001B36 EB07                   <1>       jmp    short _sm_2  ; case 1 - 19/11/2020
                                <1> _sm_1:
                                <1>       ; 26/11/2020
                                <1>       ; 19/11/2020
                                <1>
                                <1>       ; case 4
00001B38 800D[85950100]80       <1>       or     byte [p_crt_mode], 80h
                                <1>                          ; here [p_crt_mode] must be 83h
                                <1>                          ;       (for case 4)
                                <1>                          ; (because video mode 03h
                                <1>                          ; was changed before as in case 1)
                                <1>
                                <1> _sm_2: ; case 4 (jump to _sm_2) - 19/11/2020
                                <1>
                                <1>       ; 19/11/2020
                                <1>       ; case 3
                                <1>       ; If '_set_mode' procedure is called for video mode 3
                                <1>       ;     while video mode is not 3 and if there is video
                                <1>       ;     page backup for video mode 3, all (of 8) mode 3
                                <1>       ;     video pages will be restored from 98000h.
                                <1>
00001B3F A2[9A6E0000]           <1>       mov    [CRT_MODE], al  ; save mode in global variable
                                <1> _sm_3:
                                <1>       ; 30/07/2016
                                <1>       ; 26/07/2016
                                <1>       ; 25/07/2016
                                <1>       ; set_mode_vga:
                                <1>       ; 18/07/2016
                                <1>       ; 14/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; 04/07/2016
                                <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; /// video mode 13h ///
                                <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', 'vgatables.h'
                                <1>       ;
                                <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
                                <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                                <1>       ;
00001B44 88C4                   <1>       mov    ah, al
00001B46 B910000000             <1>       mov    ecx, vga_mode_count
00001B4B BE[B66E0000]           <1>       mov    esi, vga_modes
00001B50 31DB                   <1>       xor    ebx, ebx
                                <1> _sm_4:
00001B52 AC                     <1>       lodsb
00001B53 38C4                   <1>       cmp    ah, al
00001B55 7406                   <1>       je     short _sm_5
00001B57 FEC3                   <1>       inc    bl
00001B59 E2F7                   <1>       loop   _sm_4
                                <1>
                                <1>       ; UNIMPLEMENTED VIDEO MODE !
                                <1>       ;xor   eax, eax
                                <1>         ;mov [video_eax], eax ; 0
                                <1>
                                <1>       ; 26/11/2020
00001B5B F9                     <1>       stc    ; unimplemented video mode ! (cf=1)
                                <1>
00001B5C C3                     <1>       retn
                                <1>
                                <1> ;----- eBX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
                                <1>
                                <1> _sm_5:      ; 25/07/2016
                                <1>       ;mov   esi, ebx
                                <1>       ;add   esi, vga_memmodel
                                <1>       ;mov   al, [esi]
                                <1>       ; 19/11/2020
00001B5D 8A83[066F0000]         <1>       mov    al, [ebx+vga_memmodel]
00001B63 A2[9E950100]           <1>       mov    [VGA_MTYPE], al
                                <1>
00001B68 89DF                   <1>       mov    edi, ebx
00001B6A 81C7[166F0000]         <1>       add    edi, vga_dac_s
00001B70 C0E302                 <1>       shl    bl, 2 ; byte -> dword
00001B73 81C3[C66E0000]         <1>       add    ebx, vga_mode_tbl_ptr
                                <1>
                                <1>       ;mov   dword [VGA_BASE], 0B8000h
                                <1>       ;cmp   ah, 0Dh ; [CRT_MODE]
                                <1>       ;jb    short M9
                                <1>       ;mov   dword [VGA_BASE], 0A0000h
                                <1> ;M9:
00001B79 8B33                   <1>       mov    esi, [ebx]
00001B7B 89F3                   <1>       mov    ebx, esi
00001B7D 83C614                 <1>       add    esi, vga_p_cm_pos ; ebx + 20
00001B80 668B06                 <1>       mov    ax, [esi]       ; get the cursor mode from the table
00001B83 66A3[B36E0000]         <1>       mov    [CURSOR_MODE], ax ; save cursor mode (initial value)
                                <1>       ; al = 6, ah = 7
                                <1>       ; al = 0Dh, ah = 0Eh ; 25/07/2016
00001B89 E8A6020000             <1>       call   cursor_shape_fix
                                <1>       ; al = 14, ah = 15 (If [CHAR_HEIGHT] = 16)
00001B8E 668906                 <1>       mov    [esi], ax
                                <1>
```

```
4366 00001B91 56                  <1>       push   esi ; *
4367                              <1>
4368 00001B92 8A25[A16E0000]      <1>       mov    ah, [VGA_MODESET_CTL]
4369 00001B98 80E408              <1>       and    ah, 8 ; default palette loading ?
4370 00001B9B 7524                <1>       jnz    short _sm_6
4371 00001B9D 66BAC603            <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK (DAC mask register)
4372 00001BA1 B0FF                <1>       mov    al, 0FFh ; PEL mask
4373 00001BA3 EE                  <1>       out    dx, al
4374 00001BA4 8A27                <1>       mov    ah, [edi] ; DAC model (selection number)
4375 00001BA6 E859100000          <1>       call   load_dac_palette
4376                              <1>       ; ecx = 0
4377 00001BAB F605[A16E0000]02    <1>       test   byte [VGA_MODESET_CTL], 2 ; gray scale summing
4378 00001BB2 740D                <1>       jz     short _sm_6
4379 00001BB4 53                  <1>       push   ebx
4380 00001BB5 29DB                <1>       sub    ebx, ebx ; sub bl, bl
4381 00001BB7 66B90001            <1>       mov   cx, 256
4382 00001BBB E897100000          <1>       call   gray_scale_summing
4383 00001BC0 5B                  <1>       pop    ebx
4384                              <1> _sm_6:
4385                              <1>       ; Reset Attribute Ctl flip-flop
4386 00001BC1 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
4387 00001BC5 EC                  <1>       in     al, dx
4388                              <1>       ; Set Attribute Ctl
4389 00001BC6 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4390 00001BC8 83C623              <1>       add    esi, 35  ; actl regs
4391 00001BCB 30E4                <1>       xor    ah, ah ; 0
4392 00001BCD 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
4393                              <1> _sm_7:
4394 00001BD1 88E0                <1>       mov    al, ah
4395 00001BD3 EE                  <1>       out    dx, al ; index
4396 00001BD4 AC                  <1>       lodsb
4397                              <1>       ; DX = 3C0h = VGAREG_ACTL_WRITE_DATA
4398 00001BD5 EE                  <1>       out    dx, al ; value
4399 00001BD6 FEC4                <1>       inc    ah
4400 00001BD8 80FC14              <1>       cmp    ah, 20 ; number of actl registers
4401 00001BDB 72F4                <1>       jb     short _sm_7
4402                              <1>       ;
4403 00001BDD 88E0                <1>       mov    al, ah ; 20
4404 00001BDF EE                  <1>       out    dx, al ; index
4405 00001BE0 28C0                <1>       sub    al, al ; 0
4406 00001BE2 EE                  <1>       out    dx, al ; value
4407                              <1>       ;
4408                              <1>       ; Set Sequencer Ctl
4409 00001BE3 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4410 00001BE5 83C605              <1>       add    esi, 5 ; sequ regs
4411                              <1>       ;
4412 00001BE8 66BAC403            <1>       mov    dx, 3C4h  ; VGAREG_SEQU_ADDRESS
4413 00001BEC EE                  <1>       out    dx, al ; 0
4414 00001BED 6642                <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
4415 00001BEF B003                <1>       mov    al, 3
4416 00001BF1 EE                  <1>       out    dx, al
4417 00001BF2 B401                <1>       mov    ah, 1
4418                              <1> _sm_8:
4419 00001BF4 88E0                <1>       mov    al, ah
4420                              <1>       ;mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
4421 00001BF6 664A                <1>       dec    dx
4422 00001BF8 EE                  <1>       out    dx, al ; index
4423 00001BF9 AC                  <1>       lodsb
4424 00001BFA 6642                <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
4425 00001BFC EE                  <1>       out    dx, al
4426 00001BFD 80FC04              <1>       cmp    ah, 4 ; number of sequ regs
4427 00001C00 7304                <1>       jnb    short _sm_9
4428 00001C02 FEC4                <1>       inc    ah
4429 00001C04 EBEE                <1>       jmp    short _sm_8
4430                              <1> _sm_9:
4431                              <1>       ; Set Grafx Ctl
4432 00001C06 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4433 00001C08 83C637              <1>       add    esi, 55 ; grdc regs
4434 00001C0B 30E4                <1>       xor    ah, ah ; 0
4435                              <1> _sm_10:
4436 00001C0D 88E0                <1>       mov    al, ah
4437 00001C0F 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4438 00001C13 EE                  <1>       out    dx, al
4439 00001C14 AC                  <1>       lodsb
4440 00001C15 6642                <1>       inc    dx ; 3CFh ; VGAREG_GRDC_DATA
4441 00001C17 EE                  <1>       out    dx, al
4442 00001C18 FEC4                <1>       inc    ah
4443 00001C1A 80FC09              <1>       cmp    ah, 9 ; number of grdc regs
4444 00001C1D 72EE                <1>       jb     short _sm_10
4445                              <1>       ;
4446                              <1>       ; Disable CRTC write protection
4447 00001C1F 66BAD403            <1>       mov    dx, 3D4h  ; VGAREG_VGA_CRTC_ADDRESS
4448                              <1>       ;mov   al, 11h
4449                              <1>       ;our   dx, al
4450                              <1>       ;inc   dx
4451                              <1>       ;sub   al, al
4452                              <1>       ;out   dx, al
4453 00001C23 66B81100            <1>       mov    ax, 11h
4454 00001C27 66EF                <1>       out    dx, ax
4455 00001C29 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4456 00001C2B 83C60A              <1>       add    esi, 10 ; crtc regs
4457                              <1>       ; ah = 0
4458                              <1> _sm_11:
4459 00001C2E 88E0                <1>       mov    al, ah
4460                              <1>       ; dx = 3D4h = VGAREG_VGA_CRTC_ADDRESS
4461 00001C30 EE                  <1>       out    dx, al ; index
4462 00001C31 AC                  <1>       lodsb
4463 00001C32 6642                <1>       inc    dx  ; VGAREG_VGA_CRTC_ADDRESS + 1
4464 00001C34 EE                  <1>       out    dx, al ; value
4465 00001C35 80FC18              <1>       cmp    ah, 24 ; number of crtc registers - 1
4466 00001C38 7306                <1>       jnb    short _sm_12
4467 00001C3A FEC4                <1>       inc    ah
4468 00001C3C 664A                <1>       dec    dx ; 3D4h
4469 00001C3E EBEE                <1>       jmp    short _sm_11
4470                              <1> _sm_12:
```

```
4471                                    <1>       ; Set the misc register
4472 00001C40 66BACC03                  <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
4473 00001C44 8A4309                    <1>       mov    al, [ebx+9] ; misc reg
4474 00001C47 EE                        <1>       out    dx, al
4475                                    <1>       ;
4476                                    <1>       ; Enable video
4477 00001C48 66BAC003                  <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
4478 00001C4C B020                      <1>       mov    al, 20h
4479 00001C4E EE                        <1>       out    dx, al   ; set bit 5 to 1
4480 00001C4F 66BADA03                  <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
4481 00001C53 EC                        <1>       in     al, dx
4482                                    <1>       ;
4483                                    <1>       ; 17/11/2020
4484                                    <1>       ;cmp   byte [noclearmem], 0
4485                                    <1>       ;ja    short _sm_15
4486                                    <1>
4487 00001C54 F605[87950100]80          <1>       test   byte [noclearmem], 80h
4488 00001C5B 753E                      <1>       jnz    short _sm_15
4489                                    <1>
4490                                    <1>       ; 29/07/2016
4491 00001C5D 31C0                      <1>       xor    eax, eax
4492 00001C5F B900400000                <1>       mov    ecx, 4000h ; 16K words (32K)
4493 00001C64 803D[9E950100]02          <1>       cmp   byte [VGA_MTYPE], 2  ; CTEXT, MTEXT, CGA
4494 00001C6B 7715                      <1>       ja     short _sm_14    ; no ? (0A0000h)
4495 00001C6D BF00800B00                <1>       mov    edi, 0B8000h
4496 00001C72 7409                      <1>       je     short _sm_13 ; CGA graphics mode
4497                                    <1>       ; 08/08/2016
4498 00001C74 A3[9A950100]              <1>       mov    [VGA_INT43H], eax ; 0 ; default font
4499 00001C79 66B82007                  <1>       mov    ax, 0720h    ; CGA text mode
4500                                    <1> _sm_13:
4501 00001C7D F366AB                    <1>       rep    stosw
4502 00001C80 EB19                      <1>       jmp    short _sm_15
4503                                    <1>
4504                                    <1> _sm_14:
4505 00001C82 BF00000A00                <1>       mov    edi, 0A0000h
4506                                    <1>       ; ecx = 16384 dwords (64K)
4507 00001C87 66BAC403                  <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
4508 00001C8B B002                      <1>       mov    al, 2
4509 00001C8D EE                        <1>       out    dx, al
4510                                    <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
4511 00001C8E 6642                      <1>       inc    dx
4512 00001C90 EC                        <1>       in     al, dx ; mmask
4513                                    <1>       ;push  ax
4514                                    <1>       ; 12/04/2021
4515 00001C91 50                        <1>       push   eax
4516 00001C92 B00F                      <1>       mov    al, 0Fh ; all planes
4517 00001C94 EE                        <1>       out    dx, al
4518 00001C95 30C0                      <1>       xor    al, al ; 0
4519 00001C97 F3AB                      <1>       rep    stosd  ; ecx = 163684 (64K)
4520                                    <1>       ;pop   ax
4521                                    <1>       ; 12/04/2021
4522 00001C99 58                        <1>       pop    eax
4523 00001C9A EE                        <1>       out    dx, al  ; mmask
4524                                    <1> _sm_15:
4525                                    <1>       ; ebx = addr of params tbl for selected mode
4526                                    <1>       ; 10/08/2016
4527 00001C9B 668B03                    <1>       mov    ax, [ebx] ; num of columns, 'twidth'
4528 00001C9E A2[9C6E0000]              <1>       mov    [CRT_COLS], al
4529                                    <1>       ;; 26/07/2016
4530                                    <1>       ;; CRTC_ADDRESS = 3D4h (always)
4531                                    <1>       ;mov   ah, [ebx+1] ; num of rows, 'theightm1'
4532 00001CA3 FEC4                      <1>       inc    ah ; 09/07/2016
4533 00001CA5 8825[A26E0000]            <1>       mov    [VGA_ROWS], ah
4534                                    <1>       ; 10/08/2016
4535 00001CAB 8A4302                    <1>       mov    al, [ebx+2]
4536 00001CAE A2[9E6E0000]              <1>       mov    [CHAR_HEIGHT], al
4537                                    <1>       ; 29/07/2016
4538                                    <1>       ; length of regen buffer in bytes
4539 00001CB3 668B4B03                  <1>       mov    cx, [ebx+3] ; 'slength_l'
4540 00001CB7 66890D[88950100]          <1>       mov    [CRT_LEN], cx
4541                                    <1>       ;
4542                                    <1>       ; 27/07/2016
4543 00001CBE 30E4                      <1>       xor    ah, ah
4544 00001CC0 A0[1E890100]              <1>       mov    al, [ACTIVE_PAGE] ; may be > 0 for mode 3
4545                                    <1>       ;mul   word [CRT_LEN] ; 4096 for mode 3
4546 00001CC5 66F7E1                    <1>       mul    cx ; 29/07/2016
4547 00001CC8 66A3[0C890100]            <1>       mov    [CRT_START], ax
4548                                    <1>       ;
4549 00001CCE B060                      <1>       mov    al, 60h
4550                                    <1>       ;cmp   byte [noclearmem], 0
4551                                    <1>       ;jna   short _sm_16
4552                                    <1>       ;add   al, 80h
4553 00001CD0 0A05[87950100]            <1>       or     al, [noclearmem] ; 17/11/2020
4554                                    <1> _sm_16:
4555 00001CD6 A2[9F6E0000]              <1>       mov    [VGA_VIDEO_CTL], al
4556 00001CDB C605[A06E0000]F9          <1>       mov    byte [VGA_SWITCHES], 0F9h
4557 00001CE2 8025[A16E0000]7F          <1>       and    byte [VGA_MODESET_CTL], 7Fh
4558                                    <1>
4559 00001CE9 5E                        <1>       pop    esi ; *
4560                                    <1>
4561                                    <1>       ; 26/07/2016
4562                                    <1>       ; 07/07/2016
4563 00001CEA 668B0D[B36E0000]          <1>       mov    cx, [CURSOR_MODE] ; restore cursor mode (initial value)
4564 00001CF1 66870E                    <1>       xchg   cx, [esi] ; cl = start line, ch = end line
4565                                    <1>                        ; reset to initial value
4566 00001CF4 86E9                      <1>       xchg   ch, cl  ; ch = start line, cl = end line
4567 00001CF6 66890D[B36E0000]          <1>       mov    [CURSOR_MODE], cx ; save (fixed) cursor mode
4568                                    <1>
4569                                    <1>       ; 27/07/2016
4570 00001CFD 803D[9E950100]02          <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
4571 00001D04 7317                      <1>       jnb    short _sm_17
4572                                    <1>
4573                                    <1>       ; Set cursor shape
4574                                    <1>       ;mov   cx, 0607h
4575                                    <1>       ;call  _set_ctype
```

```
                                <1>
                                <1>       ; 29/07/2016
00001D06 B40A                   <1>       mov    ah, 10 ; 6845 register for cursor set
00001D08 E838060000             <1>       call   m16    ; output cx register
                                <1>
                                <1>       ; 25/07/2016
00001D0D 803D[9A6E0000]03       <1>       cmp     byte [CRT_MODE], 03h
00001D14 7507                   <1>       jne    short _sm_17
                                <1>       ; 26/07/2016
                                <1>
00001D16 A0[1E890100]           <1>       mov    al, [ACTIVE_PAGE]
00001D1B EB0B                   <1>       jmp    short _sm_18
                                <1> _sm_17:
                                <1>       ; Set cursor pos for page 0..7
                                <1>       ;sub   ax, ax ; eax = 0
00001D1D 29C0                   <1>       sub    eax, eax ; 17/11/2020
00001D1F BF[0E890100]           <1>       mov    edi, CURSOR_POSN
00001D24 AB                     <1>       stosd
00001D25 AB                     <1>       stosd
00001D26 AB                     <1>       stosd
00001D27 AB                     <1>       stosd
                                <1>       ;; Set active page 0
                                <1>       ;mov   [ACTIVE_PAGE], al ; 0
                                <1> _sm_18:
                                <1>       ; 29/07/2016
00001D28 803D[9E950100]02       <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
00001D2F 737F                   <1>         jnb  _sm_23
                                <1>
                                <1>       ;cmp   byte [CHAR_HEIGHT], 16
                                <1>       ;je    short _sm_19
                                <1>
                                <1>       ;; copy and activate 8x16 font
                                <1>
                                <1>       ; 26/07/2016
00001D31 B004                   <1>       mov    al, 04h
                                <1>       ;sub   bl, bl
                                <1>       ; AX = 1104H ; Load ROM 8x16 Character Set
                                <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
00001D33 E8BB160000             <1>       call   load_text_8_16_pat
                                <1>
                                <1>       ; video_func_1103h:
                                <1>       ; biosfn_set_text_block_specifier:
                                <1>       ; BL = font block selector code
                                <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
                                <1>       ; (It is as BL = 0 for TRDOS 386)
00001D38 66BAC403               <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
                                <1>       ;;mov  ah, bl
                                <1>       ;sub   ah, ah ; 0
                                <1>       ;mov   al, 03h
                                <1>       ; 19/11/2020
00001D3C 66B80300               <1>       mov    ax, 03h
00001D40 66EF                   <1>       out    dx, ax
                                <1> _sm_19:
                                <1>       ; 29/07/2016
                                <1>       ; 26/07/2016
                                <1>       ; 24/06/2016
                                <1>       ;mov   edi, 0B8000h
                                <1>       ;mov   cx, 4000h ; 16K words (32K)
                                <1>       ;
00001D42 30C0                   <1>       xor    al, al
00001D44 3805[85950100]         <1>       cmp    [p_crt_mode], al ; 0
00001D4A 7705                   <1>       ja     short _sm_20 ; 03h, 80h or 83h
                                <1>
                                <1>       ; case 1 - 19/11/2020
                                <1>       ;
                                <1>       ; If [pc_crt_mode] = 0, that means, previous mode is 03h
                                <1>       ; and current mode is not 03h (case 1)
                                <1>
                                <1>       ; 30/07/2016
                                <1>       ; 24/06/2016
                                <1>       ; TRDOS 386 (TRDOS v2) 'set mode' modification
                                <1>       ; (for multiscreen feature):
                                <1>       ; If '_set_mode' procedure is called for video mode 3
                                <1>       ;     while video mode is 3, video page will be cleared
                                <1>       ;     and cursor position of video page will be reset.
                                <1>       ; If '_set_mode' procedure is called for a video mode
                                <1>       ;     except video mode 3, while current video mode
                                <1>       ;     is 3, all video pages of mode 3 will be copied
                                <1>       ;     to 98000h address as backup, before mode change.
                                <1>       ; If '_set_mode' procedure is called for video mode 3
                                <1>       ;     while video mode is not 3 and if there is video
                                <1>       ;     page backup for video mode 3, all (of 8) mode 3
                                <1>       ;     video pages will be restored from 98000h.
                                <1>
                                <1>       ; 19/11/2020
                                <1>       ;mov   [ACTIVE_PAGE], al ; 0
                                <1>
                                <1>       ; Here,
                                <1>       ; video memory already cleared if [noclearmem] <> 80h
                                <1>
                                <1>       ;mov   ax, 0720h
                                <1>       ;mov   cx, 4000h ; 16K words (32K)
                                <1>       ;mov   edi, 0B8000h
                                <1>       ;rep   stosw
                                <1>       ;sub   al, al
                                <1>
                                <1>       ;jmp   short _sm_23
                                <1>
                                <1>       ; Set hardware side for the new active video page
                                <1>
00001D4C E9F9010000             <1>       jmp    _set_active_page ; 19/11/2020
                                <1>
                                <1> _sm_20:
                                <1>       ; 19/11/2020
                                <1>       ; case 2 or case 3 or case 4 - 19/11/2020
```

```
4681                                    <1>
4682                                    <1>     ; 19/11/2020
4683 00001D51 803D[9A6E0000]03          <1>     cmp   byte [CRT_MODE], 3  ; new video mode
4684 00001D58 754F                      <1>     jne   short _sm_22 ; al = 0 (& video mode <> 03h)
4685                                    <1>                        ; case 4 - 19/11/2020
4686                                    <1>                        ; ([p_crt_mode] = 83h)
4687                                    <1>
4688                                    <1>     ; case 2 or case 3 - 19/11/2020
4689                                    <1>
4690                                    <1>     ;movzx ebx, byte [ACTIVE_PAGE]
4691                                    <1>     ; 19/11/2020
4692 00001D5A A0[86950100]              <1>     mov   al, [p_crt_page] ; previous mode 3 active page
4693                                    <1>     ;movzx ebx, al
4694                                    <1>     ;shl   bl, 1 ; * 2
4695                                    <1>     ;add   ebx, CURSOR_POSN
4696                                    <1>
4697                                    <1>     ; 29/07/2016
4698 00001D5F F605[85950100]7F          <1>     test    byte [p_crt_mode], 7Fh ; 83h or 80h or 03h
4699 00001D66 740F                      <1>     jz    short _sm_21 ; do not restore video pages
4700                                    <1>                        ; case 2 - 19/11/2020
4701                                    <1>     ; case 3 - 19/11/2020
4702                                    <1>
4703                                    <1>     ; ([p_crt_mode] = 03h)
4704                                    <1>
4705                                    <1>     ; New video mode is 3 while current video mode is not 3
4706                                    <1>     ; (multi screen) video pages will be restored from 098000h
4707                                    <1>
4708                                    <1>     ; 19/11/2020
4709 00001D68 A2[1E890100]              <1>     mov   [ACTIVE_PAGE], al ; current mode 3 active page
4710                                    <1>
4711                                    <1>     ; 12/12/2020
4712                                    <1>     ;; restore video pages
4713                                    <1>     ;mov   esi, 98000h ; 30/07/2016
4714                                    <1>     ;mov   edi, 0B8000h
4715                                    <1>     ;mov   cx, 2000h ; 8K dwords (32K)
4716                                    <1>     ;rep   movsd
4717                                    <1>     ;
4718                                    <1>     ;; 19/11/2020
4719                                    <1>     ;mov   [p_crt_mode], cl ; reset ('case 3' end condition)
4720                                    <1>     ;
4721                                    <1>     ;; restore cursor positions
4722                                    <1>     ;mov   esi, cursor_pposn
4723                                    <1>     ;mov   edi, CURSOR_POSN
4724                                    <1>     ;;mov  ecx, 4 ; restore all cursor positions (16 bytes)
4725                                    <1>     ;mov   cl, 4
4726                                    <1>     ;rep   movsd
4727                                    <1>
4728                                    <1>     ; 12/12/2020
4729 00001D6D E89C000000               <1>     call   _restore_mode3_multiscreen
4730                                    <1>
4731                                    <1>     ;jmp   short _sm_23 ; do not clear current video pages
4732                                    <1>
4733                                    <1>     ; 19/11/2020
4734 00001D72 E9D3010000               <1>     jmp    _set_active_page
4735                                    <1>
4736                                    <1> _sm_21:
4737                                    <1>     ; 19/11/2020
4738                                    <1>     ; case 2
4739                                    <1>     ;
4740                                    <1>     ; ([p_crt_mode] = 80h)
4741                                    <1>     ;
4742                                    <1>     ; User has requested to set video mode 3 again while
4743                                    <1>     ; current video mode is 3.. that means, set mode 03h
4744                                    <1>     ; parameters again and clear video page.
4745                                    <1>     ; ('noclearmem' option effects the result)
4746                                    <1>
4747                                    <1>     ; 19/11/2020
4748 00001D77 F605[87950100]80          <1>     test   byte [noclearmem], 80h
4749 00001D7E 7529                      <1>     jnz    short _sm_22 ; 'do not clear video memory'
4750                                    <1>                         ; continue with current text
4751                                    <1>                         ; (user's option/choice)
4752                                    <1>     ; clear video page
4753                                    <1>     ;mov   cx, [CRT_LEN] ; 4096
4754                                    <1>     ;shr   cx, 1 ; 2048 ; 16/11/2020
4755 00001D80 66B82007                 <1>     mov    ax, 0720h
4756                                    <1>     ; 26/11/2020
4757 00001D84 B900080000               <1>     mov    ecx, 2048 ; 4096/2
4758 00001D89 BF00800B00               <1>     mov    edi, 0B8000h ; [crt_base]
4759 00001D8E 66033D[0C890100]         <1>     add    di, [CRT_START]
4760 00001D95 F366AB                   <1>     rep    stosw  ; FILL THE REGEN BUFFER WITH BLANKS
4761                                    <1>     ;
4762                                    <1>     ; 19/11/2020
4763 00001D98 A0[1E890100]             <1>     mov    al, [ACTIVE_PAGE] ; 0 to 7 (for video mode 3)
4764 00001D9D 0FB6D8                   <1>     movzx  ebx, al
4765 00001DA0 D0E3                     <1>     shl    bl, 1
4766 00001DA2 66898B[0E890100]         <1>     mov    [ebx+CURSOR_POSN], cx ; reset cursor position
4767                                    <1> _sm_22:
4768                                    <1>     ;mov   [p_crt_mode], al ; 0 ; reset
4769                                    <1>     ; 19/11/2020
4770                                    <1>     ;and   byte [p_crt_mode], 3 ; 83h -> 3, 80h -> 0
4771 00001DA9 8025[85950100]7F         <1>     and    byte [p_crt_mode], 7Fh ; 83h -> 3, 80h -> 0
4772                                    <1> _sm_23:
4773                                    <1>     ; al = video page number
4774                                    <1>     ; [CRT_LEN] = length of regen buffer in bytes
4775                                    <1>     ;call  _set_active_page
4776                                    <1>     ; 16/11/2020
4777 00001DB0 E995010000               <1>     jmp    _set_active_page
4778                                    <1>
4779                                    <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
4780                                    <1>     ;retn
4781                                    <1>
4782                                    <1> save_mode3_multiscreen:
4783                                    <1>     ; 12/12/2020 (TRDOS v2.0.3)
4784                                    <1>     ; save mode 03h video pages and cursor positions
4785                                    <1>     ;
```

```
4786                                    <1>       ; Modified registers: ecx (=0), esi, edi
4787                                    <1>
4788                                    <1>       ; 12/12/2020
4789                                    <1>       ; moved here from '_set_mode'
4790                                    <1>       ; 03/07/2016
4791                                    <1>       ; save video pages
4792 00001DB5 BE00800B00               <1>       mov    esi, 0B8000h
4793 00001DBA BF00800900               <1>       mov    edi, 98000h ; 30/07/2016
4794 00001DBF B900200000               <1>       mov    ecx, (0B8000h-0B0000h)/4
4795 00001DC4 F3A5                     <1>       rep    movsd
4796                                    <1>
4797 00001DC6 C605[85950100]03         <1>       mov    byte [p_crt_mode], 3 ; previous mode, backup sign
4798                                    <1>       ; 26/11/2020
4799 00001DCD 860D[1E890100]           <1>       xchg   cl, [ACTIVE_PAGE]
4800 00001DD3 880D[86950100]           <1>       mov    [p_crt_page], cl  ; save as previous active page
4801                                    <1>
4802                                    <1>       ; save cursor positions
4803 00001DD9 BE[0E890100]             <1>       mov    esi, CURSOR_POSN
4804 00001DDE BF[8A950100]             <1>       mov    edi, cursor_pposn ; cursor positions backup
4805 00001DE3 B104                     <1>       mov    cl, 4
4806 00001DE5 F3A5                     <1>       rep    movsd
4807 00001DE7 C3                       <1>       retn
4808                                    <1>
4809                                    <1> restore_mode3_multiscreen:
4810                                    <1>       ; 12/12/2020 (TRDOS v2.0.3)
4811                                    <1>       ; restore mode 03h video pages and cursor positions
4812                                    <1>       ;
4813                                    <1>       ; Input:
4814                                    <1>       ;    settings from the last 'save_mode3_multiscreen'
4815                                    <1>       ;
4816                                    <1>       ; Output:
4817                                    <1>       ;    AL = active video page = [ACTIVE_PAGE]
4818                                    <1>       ;
4819                                    <1>       ; Modified registers: al, ecx (=0), esi, edi
4820                                    <1>
4821 00001DE8 A0[86950100]             <1>       mov    al, [p_crt_page] ; previous mode 3 active page
4822 00001DED A2[1E890100]             <1>       mov    [ACTIVE_PAGE], al ; current mode 3 active page
4823                                    <1>
4824                                    <1>       ; 12/12/2020
4825                                    <1>       ; moved here from 'vesa_vbe3_pmi'
4826                                    <1>
4827                                    <1>       ; 07/12/2020
4828                                    <1>       ; restore CRT_START according to ACTIVE_PAGE
4829                                    <1>       ;mov   [CRT_START], cx ; 0
4830                                    <1>       ; 12/12/2020
4831 00001DF2 66C705[0C890100]00-      <1>       mov    word [CRT_START], 0
4831 00001DFA 00                       <1>
4832                                    <1>
4833                                    <1>       ; check active page and set it again if it is not 0
4834 00001DFB 08C0                     <1>       or     al, al
4835                                    <1>       ;;jz   short vbe3_pmi_7
4836                                    <1>       ;jz    short _restore_mode3_multiscreen
4837 00001DFD 740F                     <1>       jz     short r_m3_ms_1
4838 00001DFF 88C1                     <1>       mov    cl, al
4839                                    <1> ;vbe3_pmi_5:
4840                                    <1> r_m3_ms_0:
4841 00001E01 668105[0C890100]00-      <1>       add    word [CRT_START], 4096
4841 00001E09 10                       <1>
4842 00001E0A FEC9                     <1>       dec    cl
4843                                    <1>       ;jnz   short vbe3_pmi_5
4844 00001E0C 75F3                     <1>       jnz    short r_m3_ms_0
4845                                    <1> r_m3_ms_1:
4846                                    <1>       ; 12/12/2020
4847                                    <1>       ; moved here from '_set_mode'
4848                                    <1> _restore_mode3_multiscreen:
4849                                    <1>       ; Modified registers: ecx, esi, edi
4850                                    <1>
4851                                    <1>       ; restore video pages
4852 00001E0E BE00800900               <1>       mov    esi, 98000h ; 30/07/2016
4853 00001E13 BF00800B00               <1>       mov    edi, 0B8000h
4854                                    <1>       ;mov   cx, 2000h ; 8K dwords (32K)
4855 00001E18 B900200000               <1>       mov    ecx, 2000h
4856 00001E1D F3A5                     <1>       rep    movsd
4857                                    <1>
4858                                    <1>       ; 19/11/2020
4859 00001E1F 880D[85950100]           <1>       mov    [p_crt_mode], cl ; reset ('case 3' end condition)
4860                                    <1>
4861                                    <1>       ; restore cursor positions
4862 00001E25 BE[8A950100]             <1>       mov    esi, cursor_pposn
4863 00001E2A BF[0E890100]             <1>       mov    edi, CURSOR_POSN
4864                                    <1>       ;mov   ecx, 4 ; restore all cursor positions (16 bytes)
4865 00001E2F B104                     <1>       mov    cl, 4
4866 00001E31 F3A5                     <1>       rep    movsd
4867 00001E33 C3                       <1>       retn
4868                                    <1>
4869                                    <1> cursor_shape_fix:
4870                                    <1>       ; 12/04/2021
4871                                    <1>       ; 07/07/2016
4872                                    <1>       ; (Cursor start and cursor end line values -6,7-
4873                                    <1>       ; will be fixed depending on character height)
4874                                    <1>       ;
4875                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
4876                                    <1>       ; vgabios-0.7a (2011)
4877                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
4878                                    <1>       ; 'vgabios.c', ' biosfn_set_cursor_shape (CH,CL)'
4879                                    <1>       ;
4880                                    <1>       ; INPUT ->
4881                                    <1>       ;     AL = cursor start line (=6)
4882                                    <1>       ;     AH = cursor end line (=7)
4883                                    <1>       ; OUTPUT ->
4884                                    <1>       ;     AL = cursor start line (=14)
4885                                    <1>       ;     AH = cursor end line (=15)
4886                                    <1>       ;
4887                                    <1>       ;; if((modeset_ctl&0x01)&&(cheight>8)&&(CL<8)&&(CH<0x20))
4888                                    <1>
```

```
                                <1>       ;test  byte [VGA_MODESET_CTL], 1 ; VGA active
                                <1>       ;jz    short csf_3
00001E34 803D[9E6E0000]08       <1>       cmp    byte [CHAR_HEIGHT], 8
00001E3B 7647                   <1>       jna    short csf_3
00001E3D 80FC08                 <1>       cmp    ah, 8
00001E40 7342                   <1>       jnb    short csf_3
00001E42 3C20                   <1>       cmp    al, 20h
00001E44 733E                   <1>       jnb    short csf_3
                                <1>       ;
                                <1>       ;push  ax
                                <1>       ; 12/04/2021
00001E46 50                     <1>       push   eax
                                <1>       ; {
                                <1>       ; if(CL!=(CH+1))
00001E47 FEC0                   <1>       inc    al
00001E49 38C4                   <1>       cmp    ah, al   ; ah != al + 1
00001E4B 740F                   <1>         je     short csf_1
                                <1>       ; CH = ((CH+1) * cheight / 8) -1;
00001E4D 8A25[9E6E0000]         <1>       mov    ah, [CHAR_HEIGHT]
00001E53 F6E4                   <1>       mul    ah
00001E55 C0E803                 <1>       shr    al, 3 ; / 8
00001E58 FEC8                   <1>       dec    al ; - 1
00001E5A EB0E                   <1>       jmp    short csf_2
                                <1> csf_1:
                                <1>       ; }
                                <1>       ; else       ; ah = al + 1
                                <1>       ; {
00001E5C FEC4                   <1>       inc    ah     ; ah = ah + 1
                                <1>       ; CH = ((CL+1) * cheight / 8) - 2;
00001E5E A0[9E6E0000]           <1>       mov    al, [CHAR_HEIGHT]
00001E63 F6E4                   <1>       mul    ah
00001E65 C0E803                 <1>       shr    al, 3 ; / 8
00001E68 2C02                   <1>       sub    al, 2 ; - 2
                                <1>       ; al = 14 (if [CHAR_HEIGHT] = 16)
                                <1> csf_2:
00001E6A 880424                 <1>       mov    [esp], al
00001E6D 8A642401               <1>       mov    ah, [esp+1]
                                <1>       ; CL = ((CL+1) * cheight / 8) - 1;
00001E71 FEC4                   <1>       inc    ah
00001E73 A0[9E6E0000]           <1>       mov    al, [CHAR_HEIGHT]
00001E78 F6E4                   <1>       mul    ah
00001E7A C0E803                 <1>       shr    al, 3 ; / 8
00001E7D FEC8                   <1>       dec    al ; - 1
00001E7F 88442401               <1>       mov    [esp+1], al
                                <1>       ; ah = 15 (if [CHAR_HEIGHT] = 16)
                                <1>       ;
                                <1>       ;pop   ax
                                <1>       ; 12/04/2021
00001E83 58                     <1>       pop    eax
                                <1> csf_3:
00001E84 C3                     <1>       retn
                                <1>
                                <1> SET_CTYPE:
                                <1>       ; 12/09/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
00001E85 803D[9A6E0000]07       <1>       cmp    byte [CRT_MODE], 7
00001E8C 0F875FFCFFFF           <1>       ja     VIDEO_RETURN ; 12/09/2016
00001E92 E805000000             <1>       call   _set_ctype
00001E97 E955FCFFFF             <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.)
00001E9C 6689C8                 <1>       mov    ax, cx
00001E9F 86C4                   <1>       xchg   al, ah
                                <1>       ; AL = start line, AH = stop line
00001EA1 E88EFFFFFF             <1>       call   cursor_shape_fix
                                <1>       ; AL = start line (fixed), AH = stop line (fixed)
00001EA6 6689C1                 <1>       mov    cx, ax
00001EA9 86E9                   <1>       xchg   ch, cl
                                <1>       ; CH = start line (fixed), CL = stop line (fixed)
                                <1>       ;
00001EAB B40A                   <1>       mov    ah, 10 ; 6845 register for cursor set
00001EAD 66890D[B36E0000]       <1>       mov    [CURSOR_MODE], cx ; save in data area
                                <1>       ;call  m16    ; output cx register
                                <1>       ;retn
00001EB4 E98C040000             <1>       jmp   m16
                                <1>
                                <1> SET_CPOS:
                                <1>       ; 12/09/2016
```

```
                                <1>       ; 07/07/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
00001EB9 80FF07                 <1>       cmp    bh, 7 ; video page > 7 ; 07/07/2016
00001EBC 0F872FFCFFFF           <1>       ja     VIDEO_RETURN
                                <1>       ;
00001EC2 803D[9A6E0000]07       <1>       cmp    byte [CRT_MODE], 7
00001EC9 770A                   <1>       ja     short vga_set_cpos ; 12/09/2016
00001ECB E84A040000             <1>       call   _set_cpos
00001ED0 E91CFCFFFF             <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>
00001ED5 668915[0E890100]       <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
00001EDC E910FCFFFF             <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
00001EE1 80FF07                 <1>       cmp    bh, 7 ; video page > 7 (invalid)
00001EE4 7606                   <1>       jna    short read_cursor_1
                                <1>       ; invalid video page (input)
00001EE6 31C9                   <1>       xor    ecx, ecx ; 0
00001EE8 31D2                   <1>       xor    edx, edx ; 0
00001EEA EB15                   <1>       jmp    short read_cursor_2
                                <1> read_cursor_1:
                                <1>       ; 12/09/2016
00001EEC 803D[9A6E0000]07       <1>       cmp    byte [CRT_MODE], 7 ; vga mode
00001EF3 7727                   <1>       ja     short vga_get_cpos
                                <1>       ;
00001EF5 E815000000             <1>       call   get_cpos
00001EFA 0FB70D[B36E0000]       <1>       movzx  ecx, word [CURSOR_MODE]
                                <1> read_cursor_2:
00001F01 5D                     <1>       pop    ebp
00001F02 5F                     <1>       pop    edi
00001F03 5E                     <1>       pop    esi
00001F04 5B                     <1>       pop    ebx
00001F05 58                     <1>       pop    eax  ; DISCARD SAVED CX AND DX
00001F06 58                     <1>       pop    eax
00001F07 A1[78950100]           <1>       mov    eax, [video_eax] ; 12/05/2016
                                <1>       ;;15/01/2017
                                <1>       ;;mov  byte [intflg], 0 ; 07/01/2017
00001F0C 1F                     <1>       pop    ds
00001F0D 07                     <1>       pop    es
00001F0E CF                     <1>       iretd
                                <1>
                                <1> get_cpos:
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016
                                <1>       ; BH = Video page number (0 to 7)
                                <1>       ;
00001F0F D0E7                   <1>       shl    bh, 1 ; WORD OFFSET
00001F11 0FB6F7                 <1>       movzx  esi, bh
00001F14 0FB796[0E890100]       <1>       movzx  edx, word [esi+CURSOR_POSN]
00001F1B C3                     <1>       retn
                                <1>
                                <1> vga_get_cpos:
                                <1>       ; 12/09/2016
                                <1>       ; get cursor position (vga)
00001F1C 0FB715[0E890100]       <1>       movzx  edx, word [CURSOR_POSN] ; cursor pos for pg 0
00001F23 31C9                   <1>       xor    ecx, ecx ; Cursor Mode = 0 (invalid)
00001F25 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
00001F27 3C07               <1>       cmp   al, 7  ; > 7 = invalid video page number
                            <1>       ;ja   VIDEO_RETURN
00001F29 7715               <1>       ja    short adp_2 ; 18/11/2020
                            <1>       ;cmp  byte [CRT_MODE], 3
                            <1>       ;je   short adp_1
                            <1>       ; 18/11/2020
00001F2B 8A25[9A6E0000]     <1>       mov   ah, [CRT_MODE]
00001F31 80FC03             <1>       cmp   ah, 3
00001F34 7605               <1>       jna   short adp_1 ; mode 01h, 00h (01h), 02h (03h), 03h
00001F36 80FC07             <1>       cmp   ah, 7       ; mode 07h (03h)
00001F39 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:
00001F3B E805000000         <1>       call  set_active_page
                            <1> adp_2:
00001F40 E9ACFBFFFF         <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>       ;
00001F45 A2[1E890100]       <1>       mov   [ACTIVE_PAGE], al ; save active page value ; [ptty]
                            <1> _set_active_page:
                            <1>       ; 27/06/2015
00001F4A 0FB6D8             <1>       movzx  ebx, al
                            <1>       ;
                            <1>       ;cbw   ; 07/09/2014 (ah=0)
00001F4D 28E4               <1>       sub   ah, ah ; 09/12/2017
00001F4F 66F725[88950100]   <1>       mul   word [CRT_LEN]  ; get saved length of regen buffer
                            <1>                             ; display page times regen length
                            <1>       ; 10/12/2013
00001F56 66A3[0C890100]     <1>       mov   [CRT_START], ax ; save start address for later
00001F5C 6689C1             <1>       mov   cx, ax ; start address to cx
                            <1> _M16:
                            <1>       ;sar   cx, 1
00001F5F 66D1E9             <1>       shr   cx, 1  ; divide by 2 for 6845 handling
00001F62 B40C               <1>       mov   ah, 12 ; 6845 register for start address
00001F64 E8DC030000         <1>       call  m16
                            <1>       ;sal   bx, 1
                            <1>       ; 01/09/2014
00001F69 D0E3               <1>       shl   bl, 1  ; *2 for word offset
00001F6B 81C3[0E890100]     <1>       add   ebx, CURSOR_POSN
00001F71 668B13             <1>       mov   dx, [ebx] ; get cursor for this page
                            <1>       ; 16/01/2016
                            <1>       ;call  m18
                            <1>       ;retn
00001F74 E9B8030000         <1>       jmp   m18
                            <1>
                            <1> position:
                            <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
00001F79 0FB605[9C6E0000]   <1>       movzx  eax, byte [CRT_COLS] ; 24/06/2016
00001F80 F6E6               <1>       mul   dh      ; row value
00001F82 30F6               <1>       xor   dh, dh   ; 0
00001F84 6601D0             <1>       add   ax, dx  ; add column value to the result
00001F87 66D1E0             <1>       shl   ax, 1  ; * 2 for attribute bytes
                            <1>             ; EAX = AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
00001F8A C3                 <1>       retn
                            <1>
                            <1> find_position:
                            <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>
00001F8B 0FB6CF             <1>       movzx  ecx, bh ; video page number
00001F8E 89CE               <1>       mov   esi, ecx
00001F90 66D1E6             <1>       shl   si, 1
00001F93 668B96[0E890100]   <1>       mov   dx, [esi+CURSOR_POSN]
00001F9A 740C               <1>       jz    short p21
00001F9C 6631F6             <1>       xor   si, si
```

```
                                <1> p20:
00001F9F 660335[88950100]       <1>         add    si, [CRT_LEN] ; 24/06/2016
                                <1>         ;add   si, 80*25*2 ; add length of buffer for one page
00001FA6 E2F7                   <1>         loop   p20
                                <1> p21:
00001FA8 6621D2                 <1>         and    dx, dx
00001FAB 7407                   <1>         jz     short p22
00001FAD E8C7FFFFFF             <1>         call   position ; determine location in regen in page
00001FB2 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
00001FB4 66BADA03               <1>         mov    dx, 03DAh ; status port
                                <1>         ; cx = 0
00001FB8 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
00001FB9 38F5                   <1>         cmp    ch, dh
00001FBB 0F8730FBFFFF           <1>         ja     VIDEO_RETURN
00001FC1 38D1                   <1>         cmp    cl, dl
00001FC3 0F8728FBFFFF           <1>         ja     VIDEO_RETURN
                                <1>         ;
00001FC9 E805000000             <1>         call   _scroll_up
00001FCE E91EFBFFFF             <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>
00001FD3 E89B000000             <1>         call   test_line_count ; 16/01/2016
                                <1>
00001FD8 8A25[9A6E0000]         <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
00001FDE 80FC04                 <1>         cmp    ah, 4
00001FE1 720A                   <1>         jb     short n0
00001FE3 80FC07                 <1>         cmp    ah, 7 ; TEST FOR BW CARD
                                <1>                       ;      (80x25 text, mono)
00001FE6 7405                   <1>         je     short n0 ; same with mode 3 for TRDOS 386
00001FE8 E92D050000             <1>         jmp    GRAPHICS_UP
                                <1> n0:
                                <1>         ; 07/07/2016
00001FED 80FF07                 <1>         cmp    bh, 7 ; video page number
00001FF0 7606                   <1>         jna    short n1
00001FF2 8A3D[1E890100]         <1>         mov    bh, [ACTIVE_PAGE]
                                <1> n1:
00001FF8 88DC                   <1>         mov    ah, bl ; attribute
                                <1>         ;push  ax ; *
                                <1>         ; 12/04/2021
00001FFA 50                     <1>         push   eax
                                <1>         ;mov   esi, [CRT_BASE]
00001FFB BE00800B00             <1>         mov    esi, 0B8000h
00002000 3A3D[1E890100]         <1>         cmp    bh, [ACTIVE_PAGE]
00002006 750B                   <1>         jne    short n2
                                <1>         ;
00002008 66A1[0C890100]         <1>         mov    ax, [CRT_START]
0000200E 6601C6                 <1>         add    si, ax
00002011 EB11                   <1>         jmp    short n4
                                <1> n2:
00002013 20FF                   <1>         and    bh, bh
00002015 740D                   <1>         jz     short n4
00002017 88F8                   <1>         mov    al, bh
                                <1> n3:
00002019 660335[88950100]       <1>         add    si, [CRT_LEN]
00002020 FEC8                   <1>         dec    al
```

```
5309 00002022 75F5               <1>       jnz    short n3
5310                             <1> n4:
5311 00002024 E85B000000         <1>       call   scroll_position ; 16/01/2016
5312 00002029 7420               <1>       jz    short n6
5313                             <1>
5314 0000202B 01CE               <1>       add    esi, ecx ; from address for scroll
5315 0000202D 88F5               <1>       mov    ch, dh  ; #rows in block
5316 0000202F 28C5               <1>       sub    ch, al ; #rows to be moved
5317                             <1> n5:
5318 00002031 E88B000000         <1>       call   n10 ; 16/01/2016
5319                             <1>
5320 00002036 51                 <1>       push   ecx
5321 00002037 0FB60D[9C6E0000]   <1>       movzx  ecx, byte [CRT_COLS]
5322 0000203E 00C9               <1>       add    cl, cl
5323 00002040 01CE               <1>       add    esi, ecx  ; next line
5324 00002042 01CF               <1>       add    edi, ecx
5325 00002044 59                 <1>       pop    ecx
5326                             <1>
5327 00002045 FECD               <1>       dec    ch      ; count of lines to move
5328 00002047 75E8               <1>       jnz    short n5 ; row loop
5329                             <1>       ; ch = 0
5330 00002049 88C6               <1>       mov    dh, al  ; #rows
5331                             <1> n6:
5332                             <1>       ; attribute in ah
5333 0000204B B020               <1>       mov    al, ' '       ; fill with blanks
5334                             <1> n7:
5335 0000204D E87C000000         <1>       call   n11 ; 16/01/2016
5336                             <1>
5337 00002052 8A0D[9C6E0000]     <1>       mov    cl, [CRT_COLS]
5338 00002058 00C9               <1>       add    cl, cl
5339 0000205A 01CF               <1>       add    edi, ecx
5340                             <1>
5341 0000205C FECE               <1>       dec    dh
5342 0000205E 75ED               <1>       jnz    short n7
5343                             <1> n16:
5344 00002060 3A3D[1E890100]     <1>       cmp    bh, [ACTIVE_PAGE]
5345 00002066 750A               <1>       jne    short n8
5346                             <1>
5347                             <1>       ;cmp  byte [CRT_MODE], 7 ; is this the black and white card
5348                             <1>       ;je    short n8        ; if so, skip the mode reset
5349                             <1>
5350 00002068 A0[9B6E0000]       <1>       mov    al, [CRT_MODE_SET] ; get the value of mode set
5351 0000206D 66BAD803           <1>       mov    dx, 03D8h ; always set color card port
5352 00002071 EE                 <1>       out    dx, al
5353                             <1> n8:
5354 00002072 C3                 <1>       retn
5355                             <1>
5356                             <1> test_line_count:
5357                             <1>       ; 12/04/2021
5358                             <1>       ; 12/05/2016
5359                             <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5360                             <1>       ; 07/09/2014 (scroll_up)
5361 00002073 08C0               <1>       or     al, al
5362 00002075 740C               <1>       jz     short al_set2
5363                             <1>       ;push  dx
5364                             <1>       ; 12/04/2021
5365 00002077 52                 <1>       push   edx
5366 00002078 28EE               <1>       sub    dh, ch  ; subtract upper row from lower row number
5367 0000207A FEC6               <1>       inc    dh     ; adjust difference by 1
5368 0000207C 38C6               <1>       cmp    dh, al ; line count = amount of rows in window?
5369 0000207E 7502               <1>       jne    short al_set1 ; if not the we're all set
5370 00002080 30C0               <1>       xor    al, al ; otherwise set al to zero
5371                             <1> al_set1:
5372                             <1>       ;pop   dx
5373                             <1>       ; 12/04/2021
5374 00002082 5A                 <1>       pop    edx
5375                             <1> al_set2:
5376 00002083 C3                 <1>       retn
5377                             <1>
5378                             <1> scroll_position:
5379                             <1>       ; 12/04/2021
5380                             <1>       ; 26/06/2016
5381                             <1>       ; 30/01/2016
5382                             <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5383                             <1>       ; 07/09/2014 (scroll_up)
5384                             <1>
5385                             <1>       ; (*) [esp+4] = ax (al = line count, ah = attribute)
5386                             <1>
5387                             <1>       ;push  dx
5388                             <1>       ; 12/04/2021
5389 00002084 52                 <1>       push   edx
5390 00002085 6689CA             <1>       mov    dx, cx ; now, upper left position in DX
5391 00002088 E8ECFEFFFF         <1>       call   position
5392 0000208D 01C6               <1>       add    esi, eax
5393 0000208F 89F7               <1>       mov    edi, esi
5394                             <1>       ;pop   dx     ; lower right position in DX
5395                             <1>       ; 12/04/2021
5396 00002091 5A                 <1>       pop    edx
5397 00002092 6629CA             <1>       sub    dx, cx
5398 00002095 FEC6               <1>       inc    dh     ; dh = #rows
5399 00002097 FEC2               <1>       inc    dl     ; dl = #cols in block
5400 00002099 59                 <1>       pop    ecx    ; return address
5401                             <1>       ;pop   ax     ; * ; al = line count, ah = attribute
5402                             <1>       ; 12/04/2021
5403 0000209A 58                 <1>       pop    eax ; (*)
5404 0000209B 51                 <1>       push   ecx    ; return address
5405 0000209C 0FB7C8             <1>       movzx  ecx, ax
5406 0000209F 8A25[9C6E0000]     <1>       mov    ah, [CRT_COLS]
5407 000020A5 F6E4               <1>       mul    ah     ; determine offset to from address
5408 000020A7 6601C0             <1>       add    ax, ax  ; *2 for attribute byte
5409                             <1>       ;
5410                             <1>       ;push  ax     ; offset
5411                             <1>       ;push  dx
5412                             <1>       ; 12/04/2021
5413 000020AA 50                 <1>       push   eax    ; offset
```

```
                        <1>       push   edx
                        <1>       ;
                        <1>       ; 04/04/2014
000020AC 66BADA03       <1>       mov    dx, 3DAh ; guaranteed to be color card here
                        <1> n9:                   ; wait_display_enable
000020B0 EC             <1>       in     al, dx   ; get port
000020B1 A808           <1>       test   al, RVRT ; wait for vertical retrace
000020B3 74FB           <1>       jz     short n9 ; wait_display_enable
000020B5 B025           <1>       mov    al, 25h
000020B7 B2D8           <1>       mov    dl, 0D8h ; address control port
000020B9 EE             <1>       out    dx, al ; turn off video during vertical retrace
                        <1>       ;pop   dx     ; #rows, #cols
                        <1>       ;pop   ax     ; offset
                        <1>       ; 12/04/2021
000020BA 5A             <1>       pop    edx    ; #rows, #cols
000020BB 58             <1>       pop    eax    ; offset
000020BC 6691           <1>       xchg   ax, cx ;
                        <1>       ; ecx = offset, al = line count, ah = attribute
                        <1>       ;
000020BE 08C0           <1>       or     al, al
000020C0 C3             <1>       retn
                        <1> n10:
                        <1>       ; Move rows
000020C1 88D1           <1>       mov    cl, dl ; get # of cols to move
000020C3 56             <1>       push   esi
000020C4 57             <1>       push   edi    ; save start address
                        <1> n10r:
000020C5 66A5           <1>       movsw         ; move that line on screen
000020C7 FEC9           <1>       dec    cl
000020C9 75FA           <1>       jnz    short n10r
000020CB 5F             <1>       pop    edi
000020CC 5E             <1>       pop    esi    ; recover addresses
000020CD C3             <1>       retn
                        <1> n11:
                        <1>       ; Clear rows
                        <1>                      ; dh =  #rows
000020CE 88D1           <1>       mov    cl, dl ; get # of cols to clear
000020D0 57             <1>       push  edi     ; save address
                        <1> n11r:
000020D1 66AB           <1>       stosw         ; store fill character
000020D3 FEC9           <1>       dec    cl
000020D5 75FA           <1>       jnz    short n11r
000020D7 5F             <1>       pop    edi     ; recover address
000020D8 C3             <1>       retn
                        <1>
                        <1> SCROLL_DOWN:
                        <1>       ; 12/04/2021
                        <1>       ; 07/07/2016
                        <1>       ; 27/06/2016
                        <1>       ; 26/06/2016
                        <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> ; SCROLL DOWN
                        <1> ;     THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
                        <1> ;     BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
                        <1> ;     WITH A DEFINED CHARACTER
                        <1> ; INPUT
                        <1> ;     (AH) = CURRENT CRT MODE
                        <1> ;     (AL) = NUMBER OF LINES TO SCROLL
                        <1> ;     (CX) = UPPER LEFT CORNER OF RECION
                        <1> ;     (DX) = LOWER RIGHT CORNER OF REGION
                        <1> ;     (BH) = FILL CHARACTER
                        <1> ;     (DS) = DATA SEGMENT
                        <1> ;     (ES) = REGEN SEGMENT
                        <1> ; OUTPUT
                        <1> ;     NONE -- SCREEN IS SCROLLED
                        <1> ;--------------------------------------------
                        <1>
                        <1>       ; 07/07/2016
000020D9 38F5           <1>       cmp    ch, dh
                        <1>       ;ja    VIDEO_RETURN
000020DB 7709           <1>       ja     short _s_d_retn ; 18/11/2020
000020DD 38D1           <1>       cmp    cl, dl
                        <1>       ;ja    VIDEO_RETURN
000020DF 7705           <1>       ja     short _s_d_retn ; 18/11/2020
                        <1>       ;
000020E1 E805000000     <1>       call   _scroll_down
                        <1> _s_d_retn:
000020E6 E906FAFFFF     <1>       jmp    VIDEO_RETURN
                        <1>
                        <1> _scroll_down: ; 27/06/2016
                        <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>
                        <1>       ; !!!!
000020EB FD             <1>       std           ; DIRECTION FOR SCROLL DOWN
                        <1>       ; !!!!
000020EC E882FFFFFF     <1>       call   test_line_count ; 16/01/2016
                        <1>
000020F1 8A25[9A6E0000] <1>       mov    ah, [CRT_MODE] ; current video mode
                        <1>       ;;cmp  byte [CRT_MODE], 4
                        <1>       ;cmp   ah, 4 ; 07/07/2016
                        <1>       ;jnb   GRAPHICS_DOWN ; 26/06/2016
                        <1>       ; 18/11/2020
```

```
5519 000020F7 80FC04              <1>       cmp   ah, 4
5520 000020FA 720A                <1>       jb    short _n0
5521 000020FC 80FC07              <1>       cmp   ah, 7 ; TEST FOR BW CARD
5522                              <1>                   ;      (80x25 text, mono)
5523 000020FF 7405                <1>       je    short _n0 ; same with mode 3 for TRDOS 386
5524 00002101 E9F0070000          <1>       jmp   GRAPHICS_DOWN
5525                              <1> _n0:
5526                              <1>       ; 07/07/2016
5527 00002106 80FF07              <1>       cmp   bh, 7 ; video page number
5528 00002109 7606                <1>       jna   short n12
5529 0000210B 8A3D[1E890100]      <1>       mov   bh, [ACTIVE_PAGE]
5530                              <1>       ;
5531                              <1> n12:              ; CONTINUE_DOWN
5532 00002111 88DC                <1>       mov   ah, bl
5533                              <1>       ;push  ax     ; * ; save attribute in ah
5534                              <1>       ; 12/04/2021
5535 00002113 50                  <1>       push  eax
5536 00002114 6689D0              <1>       mov   ax, dx ; LOWER RIGHT CORNER
5537 00002117 E868FFFFFF          <1>       call  scroll_position     ; GET REGEN LOCATION
5538 0000211C 741F                <1>       jz    short n14
5539 0000211E 29CE                <1>       sub   esi, ecx  ; SI IS FROM ADDRESS
5540 00002120 88F5                <1>       mov   ch, dh  ; #rows in block
5541 00002122 28C5                <1>       sub   ch, al ; #rows to be moved
5542                              <1> n13:
5543 00002124 E898FFFFFF          <1>       call  n10    ; MOVE ONE ROW
5544                              <1>
5545 00002129 51                  <1>       push  ecx
5546 0000212A 8A0D[9C6E0000]      <1>       mov   cl, [CRT_COLS]
5547 00002130 00C9                <1>       add   cl, cl
5548 00002132 29CE                <1>       sub   esi, ecx  ; next line
5549 00002134 29CF                <1>       sub   edi, ecx
5550 00002136 59                  <1>       pop   ecx
5551                              <1>
5552 00002137 FECD                <1>       dec   ch      ; count of lines to move
5553 00002139 75E9                <1>       jnz   short n13 ; row loop
5554                              <1>       ; ch = 0
5555 0000213B 88C6                <1>       mov   dh, al  ; #rows
5556                              <1> n14:
5557                              <1>       ; attribute in ah
5558 0000213D B020                <1>       mov   al, ' '       ; fill with blanks
5559                              <1> n15:
5560 0000213F E88AFFFFFF          <1>       call  n11 ; 16/01/2016
5561                              <1>
5562 00002144 8A0D[9C6E0000]      <1>       mov   cl, [CRT_COLS]
5563 0000214A 00C9                <1>       add   cl, cl
5564 0000214C 29CF                <1>       sub   edi, ecx
5565                              <1>
5566 0000214E FECE                <1>       dec   dh
5567 00002150 75ED                <1>       jnz   short n15
5568                              <1>       ;
5569                              <1>       ; 18/11/2020
5570 00002152 FC                  <1>       cld   ; clear direction flag
5571                              <1>       ;
5572 00002153 E908FFFFFF          <1>       jmp   n16 ; 27/06/2016
5573                              <1>
5574                              <1> ;     cmp   bh, [ACTIVE_PAGE]
5575                              <1> ;     jne   short n16
5576                              <1> ;
5577                              <1> ;     ;cmp  byte [CRT_MODE], 7  ; is this the black and white card
5578                              <1> ;     ;je   short n16           ; if so, skip the mode reset
5579                              <1> ;
5580                              <1> ;     mov   al, [CRT_MODE_SET] ; get the value of mode set
5581                              <1> ;     mov   dx, 03D8h ; always set color card port
5582                              <1> ;     out   dx, al
5583                              <1> ;n16:
5584                              <1> ;     ; !!!!
5585                              <1> ;     cld          ; Clear direction flag !
5586                              <1> ;     ; !!!!
5587                              <1> ;     retn
5588                              <1>
5589                              <1> READ_AC_CURRENT:
5590                              <1>       ; 08/07/2016
5591                              <1>       ; 26/06/2016
5592                              <1>       ; 12/05/2016
5593                              <1>       ; 18/01/2016
5594                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5595                              <1>       ;
5596                              <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
5597                              <1>       ;
5598                              <1>       ; 08/07/2016
5599 00002158 803D[9A6E0000]07    <1>       cmp   byte [CRT_MODE], 7 ; 6!?
5600 0000215F 7607                <1>       jna   short read_ac_c
5601 00002161 31C0                <1>       xor   eax, eax
5602 00002163 E98EF9FFFF          <1>       jmp   _video_return
5603                              <1> read_ac_c:
5604 00002168 E805000000          <1>       call  _read_ac_current
5605                              <1>       ; 12/05/2016
5606                              <1>       ;jmp   VIDEO_RETURN
5607 0000216D E984F9FFFF          <1>       jmp   _video_return
5608                              <1>
5609                              <1> ;--------------------------------------------------------------------------
5610                              <1> ; READ_AC_CURRENT                                                    :
5611                              <1> ;     THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER AT THE CURRENT    :
5612                              <1> ;     CURSOR POSITION AND RETURNS THEM TO THE CALLER                  :
5613                              <1> ; INPUT                                                              :
5614                              <1> ;     (AH) = CURRENT CRT MODE                                        :
5615                              <1> ;     (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )                       :
5616                              <1> ;     (DS) = DATA SEGMENT                                            :
5617                              <1> ;     (ES) = REGEN SEGMENT                                           :
5618                              <1> ; OUTPUT                                                             :
5619                              <1> ;     (AL) = CHARACTER READ                                          :
5620                              <1> ;     (AH) = ATTRIBUTE READ                                          :
5621                              <1> ;--------------------------------------------------------------------------
5622                              <1>
5623                              <1> _read_ac_current:
```

```
                                <1>       ; 26/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 18/01/2016
                                <1>
00002172 8A25[9A6E0000]         <1>       mov    ah, [CRT_MODE] ; current video mode
00002178 80FC04                 <1>       cmp    ah, 4
0000217B 720A                   <1>       jb     short p10
                                <1>       ; 18/11/2020
                                <1>       ;cmp   byte [CRT_MODE], 4
                                <1>       ;jnb   GRAPHICS_READ ; 26/06/2016
                                <1>
0000217D 80FC07                 <1>       cmp    ah, 7 ; TEST FOR BW CARD (80x25 monochrome text)
00002180 7405                   <1>       je     short p10      ; same with mode 3 in TRDOS 386
00002182 E9C5080000             <1>       jmp    GRAPHICS_READ
                                <1> p10:
00002187 E8FFFDFFFF             <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
0000218C 803D[9A6E0000]03       <1>       cmp    byte [CRT_MODE], 03h ; 80x25 color text
                                <1>       ;je    short p11    ; Note: Only mode 03h and mode 01h are
                                <1>       ;inc   bl     ; 1  ;       in use by TRDOS 386 as text modes
                                <1>       ;jmp   short p14    ;       (07h, 00h and 02h are redirected)
00002193 7516                   <1>       jne    short p13
                                <1>
                                <1>       ; 05/12/2020
00002195 3A3D[1E890100]         <1>       cmp    bh, [ACTIVE_PAGE]
0000219B 750E                   <1>       jne    short p13
                                <1>
                                <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
                                <1> p11:
0000219D FB                     <1>       sti          ; enable interrupts first
                                <1>       ; 05/12/2020
0000219E 90                     <1>       nop
                                <1>       ; 18/11/2020
                                <1>       ;or    bl, bl
                                <1>       ;jnz   short p13 ; mode 01h (and mode 00h) - 40x25 color text
0000219F FA                     <1>       cli          ; block interrupts for single loop
000021A0 EC                     <1>       in     al, dx ; get status from the adapter
000021A1 A801                   <1>       test   al, RHRZ ; is horizontal retrace low
000021A3 75F8                   <1>       jnz    short p11 ; wait until it is
                                <1> p12:                ;  wait for either retrace high
000021A5 EC                     <1>       in     al, dx ; get status again
000021A6 A809                   <1>       test   al, RVRT+RHRZ ; is horizontal or vertical retrace high
000021A8 74FB                   <1>       jz     short p12 ; wait until either retrace active
                                <1> ;p14:
000021AA FB                     <1>       sti
                                <1> p13:
000021AB 81C600800B00           <1>       add    esi, 0B8000h
000021B1 668B06                 <1>       mov    ax, [esi]
                                <1>
000021B4 C3                     <1>       retn   ; 18/01/2016
                                <1>
                                <1> WRITE_AC_CURRENT:
                                <1>       ; 08/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 24/06/2016
                                <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> ; WRITE_AC_CURRENT                                                 :
                                <1> ;     THTS ROUTINE WRITES THE ATTRIBUTE AND CHARACTER               :
                                <1> ;     AT THE CURRENT CURSOR POSITION                               :
                                <1> ; INPUT                                                            :
                                <1> ;     (AH) = CURRENT CRT MODE                                      :
                                <1> ;     (BH) = DISPLAY PAGE                                          :
                                <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE                          :
                                <1> ;     (AL) = CHAR TO WRITE                                         :
                                <1> ;     (BL) = ATTRIBUTE OF CHAR TO WRITE                            :
                                <1> ;     (DS) = DATA SEGMENT                                          :
                                <1> ;     (ES) = REGEN SEGMENT                                         :
                                <1> ; OUTPUT                                                           :
                                <1> ;     DISPLAY REGEN BUFFER UPDATED                                 :
                                <1> ;--------------------------------------------------------------------
                                <1>
                                <1>       ; 08/07/2016
000021B5 803D[9A6E0000]07       <1>       cmp    byte [CRT_MODE], 7 ; 6!?
000021BC 760A                   <1>       jna    short write_ac_c
                                <1>
000021BE E8F70A0000             <1>       call   vga_write_char_attr
000021C3 E929F9FFFF             <1>       jmp   VIDEO_RETURN
                                <1>
                                <1> write_ac_c:
000021C8 E834000000             <1>       call   _write_c_current
                                <1>
000021CD 0FB6F7                 <1>       movzx  esi, bh ; video page number (0 to 7)
000021D0 889E[A36E0000]         <1>       mov    [esi+chr_attrib], bl ; color/attribute
                                <1>
000021D6 E916F9FFFF             <1>       jmp   VIDEO_RETURN
```

```
                                <1>
                                <1> WRITE_C_CURRENT:
                                <1>       ; 08/07/2016
                                <1>       ; 26/06/2016
                                <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> ; WRITE_C_CURRENT                                                  :
                                <1> ;     THIS ROUTINE WRITES THE CHARACTER AT                         :
                                <1> ;     THE CURRENT CURSOR POSITION, ATTRIBUTE UNCHANGED             :
                                <1> ; INPUT                                                            :
                                <1> ;     (AH) = CURRENT CRT MODE                                      :
                                <1> ;     (BH) = DISPLAY PAGE                                          :
                                <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE                          :
                                <1> ;     (AL) = CHAR TO WRITE                                         :
                                <1> ;     (DS) = DATA SEGMENT                                          :
                                <1> ;     (ES) = REGEN SEGMENT                                         :
                                <1> ; OUTPUT                                                           :
                                <1> ;     DISPLAY REGEN BUFFER UPDATED                                 :
                                <1> ;--------------------------------------------------------------------
                                <1>
                                <1>       ; 08/07/2016
000021DB 803D[9A6E0000]07       <1>       cmp   byte [CRT_MODE], 7 ; 6!?
000021E2 760A                   <1>       jna   short write_c_c
                                <1>
000021E4 E8D10A0000             <1>       call  vga_write_char_only
000021E9 E903F9FFFF             <1>       jmp   VIDEO_RETURN
                                <1>
                                <1> write_c_c:
                                <1>       ;and   bh, 7 ; video page number (<= 7)
000021EE 0FB6F7                 <1>       movzx  esi, bh
000021F1 8A9E[A36E0000]         <1>       mov   bl, [esi+chr_attrib]
                                <1>
000021F7 E805000000             <1>       call  _write_c_current
000021FC E9F0F8FFFF             <1>       jmp   VIDEO_RETURN
                                <1>
                                <1> _write_c_current:  ; from 'write_tty'
                                <1>       ; 12/04/2021
                                <1>       ; 18/11/2020
                                <1>       ; 26/06/2016
                                <1>       ; 24/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; 18/01/2014
                                <1>       ; 04/12/2013
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85 VIDEO DISPLAY BIOS
                                <1>
                                <1>       ; 18/11/2020
00002201 8A25[9A6E0000]         <1>       mov   ah, [CRT_MODE] ; current video mode
00002207 80FC04                 <1>       cmp   ah, 4
0000220A 720A                   <1>       jb    short p40
                                <1>
                                <1>       ;cmp  byte [CRT_MODE], 4
                                <1>       ;jnb  GRAPHICS_WRITE ; 26/06/2016
                                <1>
0000220C 80FC07                 <1>       cmp   ah, 7 ; TEST FOR BW CARD
0000220F 7405                   <1>       je    short p40
00002211 E986070000             <1>       jmp   GRAPHICS_WRITE
                                <1> p40:
                                <1>       ; al = character
                                <1>       ; bl = color/attribute
                                <1>       ; bh = video page
                                <1>       ; cx = count of characters to write
                                <1>       ;push  dx
                                <1>       ; 12/04/2021
00002216 52                     <1>       push  edx ; *
00002217 88DC                   <1>       mov   ah, bl  ; color/attribute (12/05/2016)
                                <1>       ;push  ax     ; save character & attribute/color
                                <1>       ;push  cx
                                <1>       ; 12/04/2021
00002219 50                     <1>       push  eax ; ** ; save character & attribute/color
0000221A 51                     <1>       push  ecx ; ***
0000221B E86BFDFFFF             <1>       call  find_position  ; get regen location and port address
                                <1>       ;pop  cx
                                <1>       ; 12/04/2021
00002220 59                     <1>       pop   ecx ; ***
                                <1>       ; esi = regen location
                                <1>       ; dx = status port
                                <1>       ;
00002221 81C600800B00           <1>       add   esi, 0B8000h ; 30/08/2014 (crt_base)
                                <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 p44    ; 26/06/2016
                                <1>
                                <1>       ; 05/12/2020
                                <1>       ; 18/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ;xor  bl, bl ; 0
00002227 803D[9A6E0000]03       <1>       cmp   byte [CRT_MODE], 03h ; 80x25 color text
                                <1>       ;je   short p41    ; Note: Only mode 03h and mode 01h are
                                <1>       ;inc  bl     ; 1   ;       in use by TRDOS 386 as text modes
                                <1>       ;jmp  short p43    ;       (07h, 00h and 02h are redirected)
                                <1>       ; 05/12/2020
0000222E 751A                   <1>       jne   short p44
                                <1> p46:
                                <1>       ; 05/12/2020
```

```
5834 00002230 3A3D[1E890100]      <1>       cmp    bh, [ACTIVE_PAGE]
5835 00002236 7512                <1>       jne    short p44
5836                              <1>
5837                              <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
5838                              <1> p41:
5839                              <1>       ; 05/12/2020
5840 00002238 FB                  <1>       sti          ; enable interrupts first
5841 00002239 90                  <1>       nop
5842                              <1>       ; 18/11/2020
5843                              <1>       ;or    bl, bl
5844                              <1>       ;jnz   short p44 ; mode 01h (and mode 00h) - 40x25 color text
5845 0000223A FA                  <1>       cli          ; block interrupts for single loop
5846 0000223B EC                  <1>       in     al, dx ; get status from the adapter
5847 0000223C A808                <1>       test   al, RVRT ; check for vertical retrace first
5848 0000223E 7509                <1>       jnz    short p43 ; Do fast write now if vertical retrace
5849 00002240 A801                <1>       test   al, RHRZ ; is horizontal retrace low
5850 00002242 75F4                <1>       jnz    short p41 ; wait until it is
5851                              <1> p42:             ;  wait for either retrace high
5852 00002244 EC                  <1>       in     al, dx ; get status again
5853 00002245 A809                <1>       test   al, RVRT+RHRZ ; is horizontal or vertical retrace high
5854 00002247 74FB                <1>       jz     short p42 ; wait until either retrace active
5855                              <1> p43:
5856 00002249 FB                  <1>       sti
5857                              <1> p44:
5858 0000224A 668B0424            <1>       mov    ax, [esp] ; restore the character (al) & attribute (ah)
5859 0000224E 668906              <1>       mov    [esi], ax
5860                              <1>
5861 00002251 6649                <1>       dec    cx
5862 00002253 740D                <1>       jz     short p45
5863                              <1>
5864 00002255 46                  <1>       inc    esi
5865 00002256 46                  <1>       inc    esi
5866                              <1>
5867                              <1>       ; 05/12/2020
5868 00002257 803D[9A6E0000]03    <1>       cmp    byte [CRT_MODE], 03h
5869 0000225E 75EA                <1>       jne    short p44
5870                              <1>       ;jmp   short p41
5871 00002260 EBCE                <1>       jmp    short p46
5872                              <1> p45:
5873                              <1>       ;pop   ax
5874                              <1>       ;pop   dx
5875                              <1>       ; 12/04/2021
5876 00002262 58                  <1>       pop    eax ; **
5877 00002263 5A                  <1>       pop    edx ; *
5878 00002264 C3                  <1>       retn
5879                              <1>
5880                              <1> ; 09/07/2016
5881                              <1> ; 26/06/2016
5882                              <1> ; 24/06/2016
5883                              <1> ; 12/05/2016
5884                              <1> ; 18/01/2016
5885                              <1> ; 16/01/2016 - TRDOS 386 (TRDOS v2.0)
5886                              <1> ; 30/06/2015
5887                              <1> ; 27/06/2015
5888                              <1> ; 11/03/2015
5889                              <1> ; 02/09/2014
5890                              <1> ; 30/08/2014
5891                              <1> ; VIDEO FUNCTIONS
5892                              <1> ; (write_tty - Retro UNIX 8086 v1 - U9.ASM, 01/02/2014)
5893                              <1>
5894                              <1> WRITE_TTY:
5895                              <1>       ; 09/12/2017
5896                              <1>       ; 09/07/2016
5897                              <1>       ; 01/07/2016
5898                              <1>       ; 26/06/2016
5899                              <1>       ; 24/06/2016
5900                              <1>       ; 13/05/2016
5901                              <1>       ; 12/05/2016
5902                              <1>       ; 30/01/2016
5903                              <1>       ; 18/01/2016
5904                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5905                              <1>       ; 13/08/2015
5906                              <1>       ; 02/09/2014
5907                              <1>       ; 30/08/2014 (Retro UNIX 386 v1 - beginning)
5908                              <1>       ; 01/02/2014 (Retro UNIX 8086 v1 - last update)
5909                              <1>       ; 03/12/2013 (Retro UNIX 8086 v1 - beginning)
5910                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
5911                              <1>       ;
5912                              <1>       ; INPUT -> AL = Character to be written
5913                              <1>       ;         BL = Color (Forecolor, Backcolor)
5914                              <1>       ;         BH = Video Page (0 to 7)
5915                              <1>
5916                              <1>       ; 09/07/2016
5917 00002265 803D[9A6E0000]07    <1>         cmp     byte [CRT_MODE], 7
5918 0000226C 760A                <1>       jna    short write_tty_cga
5919                              <1>
5920 0000226E E8320D0000          <1>       call   vga_write_teletype
5921 00002273 E979F8FFFF          <1>       jmp    VIDEO_RETURN
5922                              <1>
5923                              <1> write_tty_cga:
5924                              <1>       ; 13/05/2016
5925                              <1>       ;call  _write_tty
5926                              <1>       ; 01/07/2016
5927 00002278 E818000000          <1>       call   _write_tty_m3
5928 0000227D E96FF8FFFF          <1>       jmp    VIDEO_RETURN
5929                              <1>
5930                              <1> RVRT  equ    00001000b    ; VIDEO VERTICAL RETRACE BIT
5931                              <1> RHRZ  equ    00000001b    ; VIDEO HORIZONTAL RETRACE BIT
5932                              <1>
5933                              <1> ; Derived from "WRITE_TTY" procedure of IBM "pc-at" rombios source code
5934                              <1> ; (06/10/1985), 'video.asm', INT 10H, VIDEO_IO
5935                              <1> ;
5936                              <1> ; 06/10/85  VIDEO DISPLAY BIOS
5937                              <1> ;
5938                              <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>
00002282 FA                     <1>       cli    ; disable interrupts
                                <1>       ;
                                <1>       ; 01/09/2014
00002283 803D[9A6E0000]03       <1>       cmp    byte [CRT_MODE], 3
0000228A 7409                   <1>       je     short _write_tty_m3
                                <1>       ;
                                <1> set_mode_3:
0000228C 53                     <1>       push   ebx
0000228D 50                     <1>       push   eax
                                <1>       ;call  _set_mode
                                <1>       ; 18/11/2020
0000228E E86DF8FFFF             <1>       call   set_txt_mode  ; set video mode to 03h
00002293 58                     <1>       pop    eax
00002294 5B                     <1>       pop    ebx
                                <1>       ;
                                <1> _write_tty_m3: ; 24/06/2016 (m3 -> _write_tty_m3)
00002295 0FB6F7                 <1>       movzx  esi, bh ; 12/05/2016
00002298 66D1E6                 <1>       shl    si, 1
0000229B 81C6[0E890100]         <1>       add    esi, CURSOR_POSN
000022A1 668B16                 <1>       mov    dx, [esi]
                                <1>       ;
                                <1>       ; dx now has the current cursor position
                                <1>       ;
000022A4 3C0D                   <1>       cmp    al, 0Dh      ; CR   ; is it carriage return or control character
000022A6 763E                   <1>       jbe    short u8
                                <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>       ;
000022A8 66B90100               <1>       mov    cx, 1  ; 24/06/2016
                                <1>       ; cx = count of characters to write
                                <1>       ;
000022AC E850FFFFFF             <1>       call   _write_c_current ; 16/01/2015
                                <1>       ;
                                <1>       ; position the cursor for next char
000022B1 FEC2                   <1>       inc    dl           ; next column
000022B3 3A15[9C6E0000]         <1>       cmp    dl, [CRT_COLS]  ; test for column overflow
000022B9 755F                   <1>       jne    _set_cpos
000022BB B200                   <1>       mov    dl, 0        ; column = 0
                                <1> u10:                      ; (line feed found)
000022BD 80FE18                 <1>       cmp    dh, 25-1     ; check for last row
000022C0 7220                   <1>       jb     short u6
                                <1>       ;
                                <1>       ; scroll required
                                <1> u1:
                                <1>       ; SET CURSOR POSITION (04/12/2013)
000022C2 E853000000             <1>       call   _set_cpos
                                <1>       ;
                                <1>       ; determine value to fill with during scroll
                                <1> u2:
                                <1>       ; bh = video page number
                                <1>       ;
000022C7 E8A6FEFFFF             <1>       call   _read_ac_current ; 18/01/2016
                                <1>       ;
                                <1>       ; al = character, ah = attribute
                                <1>       ; bh = video page number
                                <1>       ; 18/11/2020
000022CC 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
```

```
6044                                    <1>       ;;mov  dl, 79
6045                                    <1>
6046                                    <1>       ;;call scroll_up    ; 04/12/2013
6047                                    <1>       ;;; 11/03/2015
6048                                    <1>       ; 02/09/2014
6049                                    <1>       ;;;mov cx, [crt_ulc] ; Upper left corner  (0000h)
6050                                    <1>       ;;;mov dx, [crt_lrc] ; Lower right corner (184Fh)
6051                                    <1>       ; 11/03/2015
6052 000022CE 6629C9                   <1>       sub    cx, cx
6053                                    <1>       ;mov   dx, 184Fh ; dl = 79 (column), dh = 24 (row)
6054                                    <1>       ; 18/11/2020
6055 000022D1 B618                     <1>       mov    dh, 25-1
6056 000022D3 8A15[9C6E0000]           <1>       mov    dl, [CRT_COLS]
6057 000022D9 FECA                     <1>       dec    dl
6058                                    <1>       ;
6059 000022DB B001                     <1>       mov    al, 1        ; scroll 1 line up
6060                                    <1>              ; ah = attribute
6061                                    <1>       ;mov   bl, al ; 12/05/2016
6062 000022DD E9F1FCFFFF               <1>       jmp    _scroll_up   ; 16/01/2016
6063                                    <1> ;u4:
6064                                    <1>       ;;int  10h          ; video-call return
6065                                    <1>                           ; scroll up the screen
6066                                    <1>                           ; tty return
6067                                    <1> ;u5:
6068                                    <1>       ;retn               ; return to the caller
6069                                    <1>
6070                                    <1> u6:                       ; set-cursor-inc
6071 000022E2 FEC6                     <1>       inc    dh           ; next row
6072                                    <1>                           ; set cursor
6073                                    <1> ;u7:
6074                                    <1>       ;;mov  ah, 02h
6075                                    <1>       ;;jmp  short u4     ; establish the new cursor
6076                                    <1>       ;call  _set_cpos
6077                                    <1>       ;jmp   short u5
6078 000022E4 EB34                     <1>       jmp     _set_cpos
6079                                    <1>
6080                                    <1>       ; check for control characters
6081                                    <1> u8:
6082 000022E6 7430                     <1>       je     short u9
6083 000022E8 3C0A                     <1>       cmp    al, 0Ah             ; is it a line feed (0Ah)
6084 000022EA 74D1                     <1>       je     short u10
6085 000022EC 3C07                     <1>       cmp    al, 07h      ; is it a bell
6086 000022EE 747C                     <1>       je     short u11
6087 000022F0 3C08                     <1>       cmp    al, 08h             ; is it a backspace
6088                                    <1>       ;jne   short u0
6089 000022F2 741C                     <1>       je     short bs     ; 12/12/2013
6090                                    <1>       ; 12/12/2013 (tab stop)
6091 000022F4 3C09                     <1>       cmp    al, 09h             ; is it a tab stop
6092 000022F6 75B0                     <1>       jne    short u0
6093 000022F8 88D0                     <1>       mov    al, dl
6094                                    <1>       ;cbw
6095 000022FA 30E4                     <1>       xor    ah, ah ; 09/12/2017
6096 000022FC B108                     <1>       mov    cl, 8
6097 000022FE F6F1                     <1>       div    cl
6098 00002300 28E1                     <1>       sub    cl, ah
6099                                    <1> ts:
6100                                    <1>       ; 02/09/2014
6101                                    <1>       ; 01/09/2014
6102                                    <1>       ;mov   al, 20h
6103                                    <1> tsloop:
6104                                    <1>       ;push  cx
6105                                    <1>       ; 12/04/2021
6106 00002302 51                       <1>       push   ecx ; *
6107                                    <1>       ; 18/11/2020
6108                                    <1>       ;push  ax
6109                                    <1>       ;;mov  bh, [ACTIVE_PAGE]
6110                                    <1>       ; 05/12/2020
6111                                    <1>       ;push  bx
6112 00002303 B020                     <1>       mov    al, 20h ; al = space (blank)
6113                                    <1>                      ; bl = color/attribute
6114 00002305 E88BFFFFFF               <1>       call   _write_tty_m3 ; 24/06/2016 (m3 -> _write_tty_m3)
6115                                    <1>       ; 05/12/2020
6116                                    <1>       ; bx is preserved in '_write_tty_m3'
6117                                    <1>       ; 18/11/2020
6118                                    <1>       ;pop   bx  ; BL = color/attribute, Bh = video page
6119                                    <1>       ;pop   ax  ; AL = character
6120                                    <1>       ; 12/04/2021
6121                                    <1>       ;pop   cx
6122 0000230A 59                       <1>       pop    ecx ; *
6123 0000230B FEC9                     <1>       dec    cl
6124 0000230D 75F3                     <1>       jnz    short tsloop
6125 0000230F C3                       <1>       retn
6126                                    <1> bs:
6127                                    <1>       ; back space found
6128                                    <1>
6129 00002310 08D2                     <1>       or     dl, dl              ; is it already at start of line
6130                                    <1>       ;je    short u7     ; set_cursor
6131 00002312 7406                     <1>       jz     short _set_cpos
6132 00002314 664A                     <1>       dec    dx                  ; no -- just move it back
6133                                    <1>       ;jmp   short u7
6134 00002316 EB02                     <1>       jmp    short _set_cpos
6135                                    <1>
6136                                    <1>       ; carriage return found
6137                                    <1> u9:
6138 00002318 B200                     <1>       mov    dl, 0        ; move to first column
6139                                    <1>       ;jmp   short u7
6140                                    <1>       ;jmp   short _set_cpos ; 30/01/2016
6141                                    <1>
6142                                    <1>       ; line feed found
6143                                    <1> ;u10:
6144                                    <1> ;     cmp    dh, 25-1     ; bottom of screen
6145                                    <1> ;     jne    short u6     ; no, just set the cursor
6146                                    <1> ;     jmp    u1           ; yes, scroll the screen
6147                                    <1>
6148                                    <1> _set_cpos:
```

```
6149                                    <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
6150                                    <1>       ; 27/06/2015
6151                                    <1>       ; 01/09/2014
6152                                    <1>       ; 30/08/2014 (Retro UNIX 386 v1)
6153                                    <1>       ;
6154                                    <1>       ; 04/12/2013 - 12/12/2013 (Retro UNIX 8086 v1)
6155                                    <1>       ;
6156                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
6157                                    <1>       ;
6158                                    <1> ;-----------------------------------------------
6159                                    <1> ; SET_CPOS
6160                                    <1> ;     THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE
6161                                    <1> ;     NEW X-Y VALUES PASSED
6162                                    <1> ; INPUT
6163                                    <1> ;     DX - ROW,COLUMN OF NEW CURSOR
6164                                    <1> ;     BH - DISPLAY PAGE OF CURSOR
6165                                    <1> ; OUTPUT
6166                                    <1> ;     CURSOR ID SET AT 6845 IF DISPLAY PAGE IS CURRENT DISPLAY
6167                                    <1> ;-----------------------------------------------
6168                                    <1>       ;
6169 0000231A BE[0E890100]             <1>       mov    esi, CURSOR_POSN
6170 0000231F 0FB6C7                   <1>       movzx  eax, bh      ; BH = video page number
6171                                    <1> ;     or     al, al
6172                                    <1> ;     jz     short _set_cpos_0
6173 00002322 D0E0                     <1>       shl    al, 1    ; word offset
6174 00002324 01C6                     <1>       add    esi, eax
6175                                    <1> ;_set_cpos_0:
6176 00002326 668916                   <1>       mov    [esi], dx ; save the pointer
6177 00002329 383D[1E890100]           <1>       cmp    [ACTIVE_PAGE], bh
6178 0000232F 7532                     <1>       jne    short m17
6179                                    <1>       ;call  m18    ; CURSOR SET
6180                                    <1> ;m17:               ; SET_CPOS_RETURN
6181                                    <1>       ; 01/09/2014
6182                                    <1> ;     retn
6183                                    <1>             ; DX  = row/column
6184                                    <1> m18:
6185 00002331 E843FCFFFF               <1>       call   position ; determine location in regen buffer
6186 00002336 668B0D[0C890100]         <1>       mov    cx, [CRT_START]
6187 0000233D 6601C1                   <1>       add    cx, ax  ; add char position in regen buffer
6188                                    <1>                      ; to the start address (offset) for this page
6189 00002340 66D1E9                   <1>       shr    cx, 1  ; divide by 2 for char only count
6190 00002343 B40E                     <1>       mov    ah, 14 ; register number for cursor
6191                                    <1>       ;call  m16    ; output value to the 6845
6192                                    <1>       ;retn
6193                                    <1>
6194                                    <1>       ;-----THIS ROUTINE OUTPUTS THE CX REGISTER
6195                                    <1>       ;     TO THE 6845 REGISTERS NAMED IN (AH)
6196                                    <1> m16:
6197 00002345 FA                       <1>       cli
6198                                    <1>       ;mov   dx, [addr_6845] ; address register
6199 00002346 66BAD403                 <1>       mov    dx, 03D4h ; I/O address of color card
6200 0000234A 88E0                     <1>       mov    al, ah ; get value
6201 0000234C EE                       <1>       out    dx, al ; register set
6202 0000234D 6642                     <1>       inc    dx     ; data register
6203 0000234F EB00                     <1>       jmp    $+2    ; i/o delay
6204 00002351 88E8                     <1>       mov    al, ch ; data
6205 00002353 EE                       <1>       out    dx, al
6206 00002354 664A                     <1>       dec    dx
6207 00002356 88E0                     <1>       mov    al, ah
6208 00002358 FEC0                     <1>       inc    al     ; point to other data register
6209 0000235A EE                       <1>       out    dx, al ; set for second register
6210 0000235B 6642                     <1>       inc    dx
6211 0000235D EB00                     <1>       jmp    $+2    ; i/o delay
6212 0000235F 88C8                     <1>       mov    al, cl ; second data value
6213 00002361 EE                       <1>       out    dx, al
6214 00002362 FB                       <1>       sti
6215                                    <1> m17:
6216 00002363 C3                       <1>       retn
6217                                    <1>
6218                                    <1> _beep:
6219                                    <1>       ; 12/02/2021 (TRDOS v2.0.3)
6220 00002364 FA                       <1>       cli
6221 00002365 E811000000               <1>       call   beep
6222 0000236A FB                       <1>       sti
6223 0000236B C3                       <1>       retn
6224                                    <1>
6225                                    <1> beeper:
6226                                    <1>       ; 04/08/2016
6227                                    <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
6228                                    <1>       ; 30/08/2014 (Retro UNIX 386 v1)
6229                                    <1>       ; 18/01/2014
6230                                    <1>       ; 03/12/2013
6231                                    <1>       ; bell found
6232                                    <1> u11:
6233 0000236C FB                       <1>       sti
6234 0000236D 3A3D[1E890100]           <1>       cmp    bh, [ACTIVE_PAGE]
6235 00002373 7553                     <1>       jne    short u12     ; Do not sound the beep
6236                                    <1>                            ; if it is not written on the active page
6237                                    <1> beeper_gfx: ; 04/08/2016
6238 00002375 66B93305                 <1>       mov    cx, 1331      ; divisor for 896 hz tone
6239 00002379 B31F                     <1>       mov    bl, 31        ; set count for 31/64 second for beep
6240                                    <1>       ;call  beep          ; sound the pod bell
6241                                    <1>       ;jmp   short u5      ; tty_return
6242                                    <1>       ;retn
6243                                    <1>
6244                                    <1> TIMER equ    040h         ; 8254 TIMER - BASE ADDRESS
6245                                    <1> PORT_B equ   061h         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
6246                                    <1> GATE2 equ    00000001b    ; TIMER 2 INPUT CATE CLOCK BIT
6247                                    <1> SPK2  equ    00000010b    ; SPEAKER OUTPUT DATA ENABLE BIT
6248                                    <1>
6249                                    <1> beep:
6250                                    <1>       ; 12/02/2021
6251                                    <1>       ; 07/02/2015
6252                                    <1>       ; 30/08/2014 (Retro UNIX 386 v1)
6253                                    <1>       ; 18/01/2014
```

```
                        <1>       ; 03/12/2013
                        <1>       ;
                        <1>       ; TEST4.ASM - 06/10/85  POST AND BIOS UTILITY ROUTINES
                        <1>       ;
                        <1>       ; ROUTINE TO SOUND THE BEEPER USING TIMER 2 FOR TONE
                        <1>       ;
                        <1>       ; ENTRY:
                        <1>       ;    (BL) = DURATION COUNTER ( 1 FOR 1/64 SECOND )
                        <1>       ;    (CX) = FREQUENCY DIVISOR (1193180/FREQUENCY) (1331 FOR 886 HZ)
                        <1>       ; EXIT:                          :
                        <1>       ;    (AX),(BL),(CX) MODIFIED.
                        <1>
0000237B 9C             <1>       pushfd  ; 18/01/2014      ; save interrupt status
0000237C FA             <1>       cli                 ; block interrupts during update
0000237D B0B6           <1>       mov    al, 10110110b; select timer 2, lsb, msb binary
0000237F E643           <1>       out    TIMER+3, al  ; write timer mode register
00002381 EB00           <1>       jmp    $+2          ; I/O delay
00002383 88C8           <1>       mov    al, cl       ; divisor for hz (low)
00002385 E642           <1>       out    TIMER+2,AL   ; write timer 2 count - lsb
00002387 EB00           <1>       jmp    $+2          ; I/O delay
00002389 88E8           <1>       mov    al, ch       ; divisor for hz (high)
0000238B E642           <1>       out    TIMER+2, al  ; write timer 2 count - msb
0000238D E461           <1>       in     al, PORT_B   ; get current setting of port
0000238F 88C4           <1>       mov    ah, al       ; save that setting
00002391 0C03           <1>       or     al, GATE2+SPK2 ; gate timer 2 and turn speaker on
00002393 E661           <1>       out    PORT_B, al   ; and restore interrupt status
                        <1>       ; 12/02/2021
00002395 9D             <1>       popfd  ; 18/01/2014
                        <1>       ;sti
                        <1> g7:                       ; 1/64 second per count (bl)
00002396 B90B040000     <1>       mov    ecx, 1035    ; delay count for 1/64 of a second
0000239B E829000000     <1>       call   waitf        ; go to beep delay 1/64 count
000023A0 FECB           <1>       dec    bl           ; (bl) length count expired?
000023A2 75F2           <1>       jnz    short g7     ; no - continue beeping speaker
                        <1>       ;
000023A4 9C             <1>       pushfd ; 12/02/2021 ; save interrupt status
000023A5 FA             <1>       cli    ; 18/01/2014 ; block interrupts during update
000023A6 E461           <1>       in     al, PORT_B   ; get current port value
                        <1>       ;or    al, not (GATE2+SPK2) ; isolate current speaker bits in case
000023A8 0CFC           <1>       or    al, ~(GATE2+SPK2)
000023AA 20C4           <1>       and    ah, al       ; someone turned them off during beep
000023AC 88E0           <1>       mov    al, ah       ; recover value of port
                        <1>       ;or   al, not (GATE2+SPK2) ; force speaker data off
000023AE 0CFC           <1>       or     al, ~(GATE2+SPK2) ; isolate current speaker bits in case
000023B0 E661           <1>       out    PORT_B, al   ; and stop speaker timer
000023B2 9D             <1>       popfd  ; 12/02/2021 ; restore interrupt flag state
                        <1>       ;sti
000023B3 B90B040000     <1>       mov    ecx, 1035    ; force 1/64 second delay (short)
000023B8 E80C000000     <1>       call   waitf        ; minimum delay between all beeps
000023BD 9C             <1>       pushfd              ; save interrupt status
000023BE FA             <1>       cli                 ; block interrupts during update
000023BF E461           <1>       in     al, PORT_B   ; get current port value in case
000023C1 2403           <1>       and    al, GATE2+SPK2      ; someone turned them on
000023C3 08E0           <1>       or     al, ah       ; recover value of port_b
000023C5 E661           <1>       out    PORT_B, al   ; restore speaker status
000023C7 9D             <1>       popfd               ; restore interrupt flag state
                        <1> u12:
000023C8 C3             <1>       retn
                        <1>
                        <1> REFRESH_BIT equ 00010000b ; REFRESH TEST BIT
                        <1>
                        <1> WAITF:
                        <1> waitf:
                        <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                        <1>       ; 03/12/2013
                        <1>       ;
                        <1> ;     push ax             ; save work register (ah)
                        <1> ;waitf1:
                        <1>                           ; use timer 1 output bits
                        <1> ;     in     al, PORT_B   ; read current counter output status
                        <1> ;     and    al, REFRESH_BIT ; mask for refresh determine bit
                        <1> ;     cmp    al, ah       ; did it just change
                        <1> ;     je     short waitf1 ; wait for a change in output line
                        <1> ;     ;
                        <1> ;     mov    ah, al       ; save new lflag state
                        <1> ;     loop   waitf1       ; decrement half cycles till count end
                        <1> ;     ;
                        <1> ;     pop    ax           ; restore (ah)
                        <1> ;     retn                ; return (cx)=0
                        <1>
                        <1> ; 06/02/2015 (unix386.s <-- dsectrm2.s)
                        <1> ; 17/12/2014 (dsectrm2.s)
                        <1> ; WAITF
                        <1> ; /// IBM PC-XT Model 286 System BIOS Source Code - Test 4 - 06/10/85 ///
                        <1> ;
                        <1> ;---WAITF--------------------------------------------------------------------------
                        <1> ;     FIXED TIME WAIT ROUTINE (HARDWARE CONTROLLED - NOT PROCESSOR)
                        <1> ; ENTRY:
                        <1> ;     (CX) = COUNT OF 15.085737 MICROSECOND INTERVALS TO WAIT
                        <1> ;            MEMORY REFRESH TIMER 1 OUTPUT USED AS REFERENCE
                        <1> ; EXIT:
                        <1> ;                  AFTER (CX) TIME COUNT (PLUS OR MINUS 16 MICROSECONDS)
                        <1> ;     (CX) = 0
                        <1> ;---------------------------------------------------------------------------------
                        <1>
                        <1> ; Refresh period: 30 micro seconds (15-80 us)
                        <1> ; (16/12/2014 - AWARDBIOS 1999 - ATORGS.ASM, WAIT_REFRESH)
                        <1>
                        <1> ;WAITF:                         ; DELAY FOR (CX)*15.085737 US
                        <1>       ;PUSH  AX                 ; SAVE WORK REGISTER (AH)
                        <1>       ; 11/04/2021
000023C9 50             <1>       push   eax
                        <1>       ; 16/12/2014
                        <1>       ;shr   cx, 1              ; convert to count of 30 micro seconds
000023CA D1E9           <1>       shr    ecx, 1 ; 21/02/2015
```

```
                                <1> ;17/12/2014
                                <1> ;WAITF1:
                                <1> ;       IN      AL, PORT_B   ;061h  ; READ CURRENT COUNTER OUTPUT STATUS
                                <1> ;       AND     AL, REFRESH_BIT     ;00010000b ; MASK FOR REFRESH DETERMINE BIT
                                <1> ;       CMP     AL, AH              ; DID IT JUST CHANGE
                                <1> ;       JE      short WAITF1        ; WAIT FOR A CHANGE IN OUTPUT LINE
                                <1> ;       MOV     AH, AL              ; SAVE NEW FLAG STATE
                                <1> ;       LOOP    WAITF1              ; DECREMENT HALF CYCLES TILL COUNT END
                                <1>         ;
                                <1>         ; 17/12/2014
                                <1>         ;
                                <1>         ; Modification from 'WAIT_REFRESH' procedure of AWARD BIOS - 1999
                                <1>         ;
                                <1> ;WAIT_REFRESH:  Uses port 61, bit 4 to have CPU speed independent waiting.
                                <1> ;       INPUT:  CX = number of refresh periods to wait
                                <1> ;                       (refresh periods = 1 per 30 microseconds on most machines)
                                <1> WR_STATE_0:
000023CC E461                   <1>         IN      AL,PORT_B           ; IN AL,SYS1
000023CE A810                   <1>         TEST    AL,010H
000023D0 74FA                   <1>         JZ      SHORT WR_STATE_0
                                <1> WR_STATE_1:
000023D2 E461                   <1>         IN      AL,PORT_B           ; IN AL,SYS1
000023D4 A810                   <1>         TEST    AL,010H
000023D6 75FA                   <1>         JNZ     SHORT WR_STATE_1
000023D8 E2F2                   <1>         LOOP     WR_STATE_0
                                <1>         ;
                                <1>         ;POP    AX                  ; RESTORE (AH)
                                <1>         ; 11/04/2021
000023DA 58                     <1>         pop     eax
000023DB C3                     <1>         RETn                        ; (CX) = 0
                                <1>
                                <1> ; 09/07/2016
                                <1> ; 01/07/2016
                                <1> ; 24/06/2016
                                <1> ; 23/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; WRITE_STRING                                                              :
                                <1> ;       THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.               :
                                <1> ; INPUT                                                                     :
                                <1> ;       (AL) = WRITE STRING COMMAND  0 - 3                                  :
                                <1> ;       (BH) = DISPLAY PAGE (ACTIVE PAGE)                                   :
                                <1> ;       (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN        :
                                <1> ;       (DX) = CURSOR POSITION FOR START OF STRING WRITE                     :
                                <1> ;       (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR (AL) = 1     :
                                <1> ;       (eBP) = SOURCE STRING OFFSET                                        :
                                <1> ; OUTPUT                                                                    :
                                <1> ;       NONE                                                                :
                                <1> ;--------------------------------------------------------------------------------
                                <1>
                                <1> ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
                                <1> ; AL = 01h: Assign all characters the attribute in BL; update cursor
                                <1> ; AL = 02h: Use attributes in string; do not update cursor
                                <1> ; AL = 03h: Use attributes in string; update cursor
                                <1>
                                <1> WRITE_STRING:
                                <1>         ; 12/09/2016
                                <1>         ; 09/07/2016
                                <1>         ;cmp   byte [CRT_MODE], 7 ; 6?!
                                <1>         ;ja    VIDEO_RETURN         ; not a valid function for VGA modes
                                <1>         ;
000023DC A2[84950100]           <1>         mov    [w_str_cmd], al      ; save (AL) command
000023E1 3C04                   <1>         CMP    AL, 4                ; TEST FOR INVALID WRITE STRING OPTION
000023E3 0F8308F7FFFF           <1>         JNB    VIDEO_RETURN         ; IF OPTION INVALID THEN RETURN
                                <1>
                                <1>        ;JCXZ   VIDEO_RETURN         ; IF ZERO LENGTH STRING THEN RETURN
                                <1>
000023E9 67E362                 <1>        jcxz    P55                  ; 01/07/2016
                                <1>
                                <1>         ; 01/07/2016
                                <1>         ;and   ecx, 0FFFFh
                                <1>         ; ECX = byte count
                                <1>         ;push  ecx
000023EC 89EE                   <1>         mov    esi, ebp ; user buffer
000023EE BF00000700             <1>         mov    edi, Cluster_Buffer  ; system buffer
000023F3 E876F60000             <1>         call   transfer_from_user_buffer
                                <1>         ;pop   ecx
000023F8 0F82F3F6FFFF           <1>         jc     VIDEO_RETURN
                                <1>         ; ecx = transfer (byte) count = character count
000023FE BD00000700             <1>         mov    ebp, Cluster_Buffer
                                <1>         ; 12/09/2016
00002403 803D[9A6E0000]07       <1>         cmp    byte [CRT_MODE], 7 ; 6?!
0000240A 0F87A1000000           <1>         ja     vga_write_string
                                <1>         ;
00002410 0FB6F7                 <1>         movzx  esi, bh              ; GET CURRENT CURSOR PAGE
00002413 66D1E6                 <1>         SAL    SI, 1                ; CONVERT TO PAGE OFFSET  (SI= PAGE)
                                <1>         ; *****
00002416 66FFB6[0E890100]       <1>         PUSH   word [eSI+CURSOR_POSN] ; SAVE CURRENT CURSOR POSITION IN STACK
                                <1>
                                <1>         ;MOV   AX,0200H             ; SET NEW CURSOR POSITION
                                <1>         ;INT   10H
                                <1> P50next:
0000241D 51                     <1>         push   ecx ; ****
0000241E 53                     <1>         push   ebx ; *** ; 18/11/2020
0000241F 56                     <1>         push   esi ; **
00002420 52                     <1>         push   edx ; *
00002421 E8F4FEFFFF             <1>         call   _set_cpos
                                <1> P50:
00002426 8A4500                 <1>         MOV    AL, [eBP]            ; GET CHARACTER FROM INPUT STRING
00002429 45                     <1>         INC    eBP                  ; BUMP POINTER TO CHARACTER
                                <1>
                                <1> ;-----      TEST FOR SPECIAL CHARACTER'S
                                <1>
0000242A 3C08                   <1>         CMP    AL, 08H              ; IS IT A BACKSPACE
0000242C 7410                   <1>         JE     short P51            ; BACK_SPACE
```

```
6464 0000242E 3C0D              <1>       CMP    AL, 0Dh ; CR        ; IS IT CARRIAGE RETURN
6465 00002430 740C              <1>       JE     short P51           ; CAR_RET
6466 00002432 3C0A              <1>       CMP    AL, 0Ah ; LF        ; IS IT A LINE FEED
6467 00002434 7408              <1>       JE     short P51           ; LINE_FEED
6468                            <1>       ; 18/11/2020
6469 00002436 3C09              <1>       cmp    al, 09h                  ; is it a tab stop
6470 00002438 7404              <1>       je     short P51
6471                            <1>       ;
6472 0000243A 3C07              <1>       CMP    AL, 07h             ; IS IT A BELL
6473 0000243C 7515              <1>       JNE    short P52           ; IF NOT THEN DO WRITE CHARACTER
6474                            <1> P51:
6475                            <1>       ;MOV   AH,0EH              ; TTY_CHARACTER_WRITE
6476                            <1>       ;INT   10H                 ; WRITE TTY CHARACTER TO THE CRT
6477                            <1>
6478 0000243E E852FEFFFF        <1>       call   _write_tty_m3
6479                            <1>
6480 00002443 5A                <1>       pop    edx ; *
6481 00002444 5E                <1>       pop    esi ; **
6482                            <1>
6483 00002445 668B96[0E890100]  <1>       MOV    DX, [eSI+CURSOR_POSN] ; GET CURRENT CURSOR POSITION
6484 0000244C EB44              <1>       JMP    SHORT P54           ; SET CURSOR POSITION AND CONTINUE
6485                            <1> P55:
6486 0000244E E99EF6FFFF        <1>       JMP    VIDEO_RETURN
6487                            <1> P52:
6488 00002453 66B90100          <1>       MOV    CX, 1               ; SET CHARACTER WRITE AMOUNT TO ONE
6489 00002457 803D[84950100]02  <1>       CMP    byte [w_str_cmd], 2 ; IS THE ATTRIBUTE IN THE STRING
6490 0000245E 7204              <1>       JB     short P53           ; IF NOT THEN SKIP
6491 00002460 8A5D00            <1>       MOV    BL, [eBP]           ; ELSE GET NEW ATTRIBUTE
6492 00002463 45                <1>       INC    eBP                 ; BUMP STRING POINTER
6493                            <1> P53:
6494                            <1>       ;MOV   AH,09H              ; GOT_CHARACTER
6495                            <1>       ;INT   10H                 ; WRITE CHARACTER TO THE CRT
6496                            <1>
6497 00002464 E898FDFFFF        <1>       call   _write_c_current
6498                            <1>
6499 00002469 5A                <1>       pop    edx ; *
6500                            <1>
6501                            <1>       ; 05/12/2020
6502                            <1>       ; bx is preserved in '_write_c_current'
6503                            <1>       ; 18/11/2020
6504                            <1>       ;mov   ebx, [esp+4] ; ***
6505                            <1>
6506 0000246A 0FB6F7            <1>       movzx  esi, bh ; video page number (0 to 7)
6507 0000246D 889E[A36E0000]    <1>       mov    [esi+chr_attrib], bl ; color/attribute
6508                            <1>
6509 00002473 FEC2              <1>       INC    DL                  ; INCREMENT COLUMN COUNTER
6510 00002475 3A15[9C6E0000]    <1>       CMP    DL, [CRT_COLS]      ; IF COLS ARE WITHIN RANGE FOR THIS MODE
6511                            <1>       ;JB    short P54           ;    THEN GO TO COLUMNS SET
6512 0000247B 7214              <1>       jb     short P56 ; 05/12/2020
6513 0000247D FEC6              <1>       INC    DH                  ; BUMP ROW COUNTER BY ONE
6514 0000247F 28D2              <1>       SUB    DL, DL              ; SET COLUMN COUNTER TO ZERO
6515 00002481 80FE19            <1>       CMP    DH, 25              ; IF ROWS ARE LESS THAN 25 THEN
6516                            <1>       ;JB    short P54           ; GO TO ROWS_COLUMNS_SET
6517 00002484 720B              <1>       jb     short P56 ; 05/12/2020
6518                            <1>
6519                            <1>       ; 18/11/2020
6520                            <1>       ;MOV   AX,0E0AH            ; ELSE SCROLL SCREEN
6521                            <1>       ;INT   10H                 ; RESET ROW COUNTER TO 24
6522                            <1>
6523                            <1>       ; 18/11/2020
6524 00002486 B00A              <1>       mov    al, 0Ah      ; line feed
6525                            <1>
6526 00002488 E808FEFFFF        <1>       call   _write_tty_m3
6527                            <1>
6528 0000248D 66BA0018          <1>       mov    dx, 1800h           ; Column = 0, Row = 24
6529                            <1> P56:
6530                            <1>       ; 05/12/2020
6531                            <1>       ; 18/11/2020
6532 00002491 5E                <1>       pop    esi ; **
6533                            <1> P54:                              ; ROW_COLUMNS_SET
6534                            <1>       ;MOV   AX,0200H            ; SET NEW CURSOR POSITION COMMAND
6535                            <1>       ;INT   10H                 ; ESTABLISH NEW CURSOR POSITION
6536                            <1>
6537                            <1>       ; 18/11/2020
6538 00002492 5B                <1>       pop    ebx ; ***
6539 00002493 59                <1>       pop    ecx ; ****
6540                            <1>
6541                            <1>       ;LOOP  P50                 ; DO IT ONCE MORE UNTIL (CX) = ZERO
6542 00002494 6649              <1>       dec    cx
6543 00002496 7585              <1>       jnz    short P50next
6544                            <1>
6545 00002498 665A              <1>       POP    DX  ; *****         ; RESTORE OLD CURSOR COORDINATES
6546                            <1>
6547 0000249A F605[84950100]01  <1>       test   byte [w_str_cmd], 1 ; IF CURSOR WAS NOT TO BE MOVED
6548 000024A1 0F854AF6FFFF      <1>       JNZ    VIDEO_RETURN        ; THEN EXIT WITHOUT RESETTING OLD VALUE
6549                            <1>
6550                            <1>       ;MOV   AX,0200H            ; ELSE RESTORE OLD CURSOR POSITION
6551                            <1>       ;INT   10H
6552                            <1>                                   ; DONE - EXIT WRITE STRING
6553 000024A7 E86EFEFFFF        <1>       call   _set_cpos
6554 000024AC E940F6FFFF        <1>       JMP    VIDEO_RETURN        ; RETURN TO CALLER
6555                            <1>
6556                            <1> vga_write_string:
6557                            <1>       ; 12/09/2016 - TRDOS 386 (TRDOS v2.0)
6558                            <1>       ;
6559                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
6560                            <1>       ; vgabios-0.7a (2011)
6561                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
6562                            <1>       ; 'vgabios.c', ' biosfn_write_string'
6563                            <1>
6564                            <1>       ; INPUT                                                          :
6565                            <1>       ;      (AL) = WRITE STRING COMMAND  0 - 3                        :
6566                            <1>       ;      (BH) = DISPLAY PAGE (ACTIVE PAGE)                         :
6567                            <1>       ;      (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN :
6568                            <1>       ;      (DX) = CURSOR POSITION FOR START OF STRING WRITE           :
```

```
                                <1>       ;      (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR (AL) = 1   :
                                <1>       ;      (eBP) = SOURCE STRING OFFSET                                      :
                                <1>       ; OUTPUT                                                                 :
                                <1>       ;      NONE                                                              :
                                <1>       ;-------------------------------------------------------------------------;
                                <1>
                                <1>       ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
                                <1>       ; AL = 01h: Assign all characters the attribute in BL; update cursor
                                <1>       ; AL = 02h: Use attributes in string; do not update cursor
                                <1>       ; AL = 03h: Use attributes in string; update cursor
                                <1>
                                <1>       ; biosfn_write_string(GET_AL(),GET_BH(),GET_BL(),CX,GET_DH(),GET_DL(),ES,BP);
                                <1>       ; static void biosfn_write_string (flag,page,attr,count,row,col,seg,offset)
                                <1>
                                <1>       ; // Read curs info for the page
                                <1>       ; biosfn_get_cursor_pos(page,&dummy,&oldcurs);
                                <1>       ; bh = video page = 0
                                <1>       ;movzx esi, word [CURSOR_POSN] ; current cursor position for video page 0
                                <1>
                                <1>       ; // if row=0xff special case : use current cursor position
                                <1>       ; if(row==0xff)
                                <1>       ;  {col=oldcurs&0x00ff;
                                <1>       ;   row=(oldcurs&0xff00)>>8;
                                <1>       ;  }
                                <1>
                                <1>       ;mov   al, [w_str_cmd]
                                <1>
000024B1 80FEFF                 <1>       cmp    dh, 0FFh
000024B4 7407                   <1>       je     short vga_wstr_1 ; user current cursor position
                                <1> vga_wstr_0:
                                <1>       ; set cursor position
000024B6 668915[0E890100]       <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
                                <1> vga_wstr_1:
000024BD 66FF35[0E890100]       <1>       push   word [CURSOR_POSN] ; *
                                <1>
                                <1>       ; ebp = string offset in system buffer (user buffer was copied to)
                                <1>
                                <1>       ; while(count--!=0)
                                <1>       ;  {
                                <1>       ;   car=read_byte(seg,offset++);
                                <1>       ;   if((flag&0x02)!=0)
                                <1>       ;    attr=read_byte(seg,offset++);
                                <1>       ;    biosfn_write_teletype(car,page,attr,WITH_ATTR);
                                <1>       ;  }
                                <1>
                                <1>       ;push  eax ; **
                                <1>       ;test  al, 2
000024C4 F605[84950100]02       <1>       test   byte [w_str_cmd], 2
000024CB 751D                   <1>       jnz    short vga_wstr_3
000024CD 881D[1F890100]         <1>       mov    [ccolor], bl
                                <1> vga_wstr_2:
000024D3 51                     <1>       push   ecx
000024D4 8A4500                 <1>       mov    al, [ebp]
000024D7 E8C90A0000             <1>       call   vga_write_teletype
000024DC 59                     <1>       pop    ecx
000024DD 6649                   <1>       dec    cx
000024DF 741E                   <1>       jz     short vga_wstr_4
000024E1 45                     <1>       inc    ebp
000024E2 8A1D[1F890100]         <1>       mov    bl, [ccolor]
000024E8 EBE9                   <1>       jmp    short vga_wstr_2
                                <1> vga_wstr_3:
000024EA 51                     <1>       push   ecx
000024EB 8A4500                 <1>       mov    al, [ebp]
000024EE 45                     <1>       inc    ebp
000024EF 8A5D00                 <1>       mov    bl, [ebp]
000024F2 E8AE0A0000             <1>       call   vga_write_teletype
000024F7 59                     <1>       pop    ecx
000024F8 6649                   <1>       dec    cx
000024FA 7403                   <1>       jz     short vga_wstr_4
000024FC 45                     <1>       inc    ebp
000024FD EBEB                   <1>       jmp    short vga_wstr_3
                                <1> vga_wstr_4:
                                <1>       ; // Set back curs pos
                                <1>       ; if((flag&0x01)==0)
                                <1>       ;  biosfn_set_cursor_pos(page,oldcurs);
                                <1>       ; }
                                <1>       ;pop   eax ; **
000024FF 665A                   <1>       pop    dx ; word [CURSOR_POSN] ; *
                                <1>       ;test  al, 1
00002501 F605[84950100]01       <1>       test   byte [w_str_cmd], 1
00002508 0F85E3F5FFFF           <1>       jnz    VIDEO_RETURN
0000250E 668915[0E890100]       <1>       mov    [CURSOR_POSN], dx
00002515 E9D7F5FFFF             <1>       JMP    VIDEO_RETURN
                                <1>
                                <1> ; 07/07/2016
                                <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;--------------------------------------------------
                                <1> ;  SCROLL UP
                                <1> ;   THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
                                <1> ; ENTRY ---
                                <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
                                <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
                                <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
                                <1> ;  BH = FILL VALUE FOR BLANKED LINES
                                <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
                                <1> ;  DS = DATA SEGMENT
                                <1> ;  ES = REGEN SEGMENT
                                <1> ; EXIT --
                                <1> ;  NOTHING, THE SCREEN IS SCROLLED
                                <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> GRAPHICS_UP:
                        <1>       ; 07/07/2016
                        <1>       ;AH = Current video mode, [CRT_MODE]
0000251A 80FC07         <1>       cmp    ah, 7
0000251D 7766           <1>       ja     short vga_graphics_up
                        <1>       ;je    n0
                        <1>
0000251F 88C7           <1>       MOV    bh, al             ; save line count in BH
00002521 6689C8         <1>       MOV    AX, CX             ; GET UPPER LEFT POSITION INTO AX REG
                        <1>
                        <1> ;-----       USE CHARACTER SUBROUTINE FOR POSITIONING
                        <1> ;-----       ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                        <1>
00002524 E8D0050000     <1>       CALL   GRAPH_POSN
00002529 0FB7F8         <1>       MOVzx  eDI, AX            ; SAVE RESULT AS DESTINATION ADDRESS
                        <1>
                        <1> ;-----       DETERMINE SIZE OF WINDOW
                        <1>
0000252C 6629CA         <1>       SUB    DX, CX
0000252F 6681C20101     <1>       ADD    DX, 101h           ; ADJUST VALUES
00002534 C0E602         <1>       SAL    DH, 2              ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
                        <1>                                 ; AND EVEN/ODD ROWS
                        <1> ;-----       DETERMINE CRT MODE
                        <1>
00002537 803D[9A6E0000]06 <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
0000253E 7305           <1>         JNC     short _R7_             ; FIND_SOURCE
                        <1>
                        <1> ;-----       MEDIUM RES UP
00002540 D0E2           <1>       SAL    DL, 1              ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
00002542 66D1E7         <1>       SAL    DI, 1              ; OFFSET *2 SINCE 2 BYTES/CHAR
                        <1>
                        <1> ;-----       DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                        <1> _R7_:                           ; FIND_SOURCE
00002545 81C700800B00   <1>       add    edi, 0B8000h
0000254B C0E702         <1>       sal    bh, 2              ; multiply number of lines by 4
0000254E 7431           <1>       JZ     short _R11         ; IF ZERO, THEN BLANK ENTIRE FIELD
00002550 B050           <1>       MOV    AL, 80             ; 80 BYTES/ROW
00002552 F6E7           <1>       mul    bh                 ; determine offset to source
00002554 0FB7F0         <1>       movzx  esi, ax            ; offset to source
00002557 01FE           <1>       add    eSI, eDI           ; SET UP SOURCE
00002559 88F4           <1>       MOV    AH, DH             ; NUMBER OF ROWS IN FIELD
0000255B 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
0000255D E80B040000     <1>       CALL    _R17              ; MOVE ONE ROW
00002562 6681EEB01F     <1>       SUB    SI, 2000h-80       ; MOVE TO NEXT ROW
00002567 6681EFB01F     <1>       SUB    DI, 2000h-80
0000256C FECC           <1>       DEC    AH                 ; NUMBER OF ROWS TO MOVE
0000256E 75ED           <1>       JNZ    short _R8          ; CONTINUE TILL ALL MOVED
                        <1>
                        <1> ;-----       FILL IN THE VACATED LINE(S)
                        <1> _R9:                            ; CLEAR ENTRY
00002570 88D8           <1>       mov    al, bl             ; attribute to fill with
                        <1> _R10_:
00002572 E812040000     <1>       CALL   _R18               ; CLEAR THAT ROW
00002577 6681EFB01F     <1>       SUB    DI, 2000h-80       ; POINT TO NEXT LINE
0000257C FECF           <1>       dec    bh                 ; number of lines to fill
0000257E 75F2           <1>       JNZ    short _R10_        ; CLEAR LOOP
00002580 C3             <1>       retn                      ; EVERYYHING DONE
                        <1>
                        <1> _R11:                           ; BLANK_FIELD
00002581 88F7           <1>       mov    bh, dh             ; set blank count to everything in field
00002583 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>
00002585 88C7           <1>       mov    bh, al ; 31/07/2016
00002587 BE[BE6E0000]   <1>       mov    esi, vga_g_modes
0000258C 89F7           <1>       mov    edi, esi
0000258E 83C708         <1>       add    edi, vga_g_mode_count
                        <1> vga_g_up_0:
00002591 AC             <1>       lodsb
00002592 38E0           <1>       cmp    al, ah ; [CRT_MODE]
00002594 7405           <1>       je     short vga_g_up_1
```

```
6779 00002596 39FE              <1>       cmp    esi, edi
6780 00002598 72F7              <1>       jb     short vga_g_up_0
6781                            <1>       ;xor   bh, bh ; 31/07/2016)
6782 0000259A C3                <1>       retn   ; nothing to do
6783                            <1> vga_g_up_1:
6784 0000259B 88F8              <1>       mov    al, bh ; 31/07/2016
6785 0000259D 83C64F            <1>       add    esi, vga_g_memmodel - (vga_g_modes + 1)
6786                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6787                            <1>
6788                            <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
6789                            <1>       ; if(clr>=nbcols)clr=nbcols-1;
6790                            <1>       ; if(nblines>nbrows)nblines=0;
6791                            <1>       ; cols=clr-cul+1;
6792                            <1>
6793 000025A0 3A35[A26E0000]    <1>       cmp    dh, [VGA_ROWS]
6794 000025A6 7208              <1>       jb     short vga_g_up_2
6795 000025A8 8A35[A26E0000]    <1>       mov    dh, [VGA_ROWS]
6796 000025AE FECE              <1>       dec    dh
6797                            <1> vga_g_up_2:
6798 000025B0 3A15[9C6E0000]    <1>       cmp    dl, [CRT_COLS]  ; = [VGA_COLS]
6799 000025B6 7208              <1>       jb     short vga_g_up_3
6800 000025B8 8A15[9C6E0000]    <1>       mov    dl, [CRT_COLS]
6801 000025BE FECA              <1>       dec    dl
6802                            <1> vga_g_up_3:
6803 000025C0 3A05[A26E0000]    <1>       cmp    al, [VGA_ROWS]
6804 000025C6 7602              <1>       jna    short vga_g_up_4
6805 000025C8 28C0              <1>       sub    al, al ; 0
6806                            <1> vga_g_up_4:
6807 000025CA 88D7              <1>       mov    bh, dl ; clr
6808 000025CC 28CF              <1>       sub    bh, cl ; cul
6809 000025CE FEC7              <1>       inc    bh ; cols = clr-cul+1
6810                            <1>
6811 000025D0 20C0              <1>       and    al, al ; nblines = 0
6812 000025D2 7559              <1>       jnz    short vga_g_up_6
6813 000025D4 20ED              <1>       and    ch, ch ; rul = 0
6814 000025D6 7555              <1>       jnz    short vga_g_up_6
6815 000025D8 20C9              <1>       and    cl, cl ; cul = 0
6816 000025DA 7551              <1>       jnz    short vga_g_up_6
6817                            <1>
6818                            <1>       ;push  ax
6819                            <1>       ; 12/04/2021
6820 000025DC 50                <1>       push   eax
6821 000025DD A0[A26E0000]      <1>       mov    al, [VGA_ROWS]
6822 000025E2 FEC8              <1>       dec    al
6823 000025E4 38C6              <1>       cmp    dh, al ; rlr = nbrows-1
6824 000025E6 7545              <1>       jne    short vga_g_up_5
6825 000025E8 A0[9C6E0000]      <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
6826 000025ED FEC8              <1>       dec    al
6827 000025EF 38C2              <1>       cmp    dl, al ; clr = nbcols-1
6828                            <1>       ;jne   short vga_g_up_5
6829                            <1>       ;;pop  ax
6830                            <1>       ; 12/04/2021
6831 000025F1 58                <1>       pop    eax
6832 000025F2 7539              <1>       jne    short vga_g_up_5
6833                            <1>
6834 000025F4 66B80502          <1>       mov    ax, 0205h
6835 000025F8 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6836 000025FC 66EF              <1>       out    dx, ax
6837 000025FE A0[A26E0000]      <1>       mov    al, [VGA_ROWS]
6838 00002603 8A25[9C6E0000]    <1>       mov    ah, [CRT_COLS] ; = [VGA_COLS]
6839 00002609 F6E4              <1>       mul    ah
6840 0000260B 0FB7D0            <1>       movzx  edx, ax
6841                            <1>       ; 08/08/2016
6842 0000260E 0FB605[9E6E0000]  <1>       movzx  eax, byte [CHAR_HEIGHT]
6843 00002615 F7E2              <1>       mul    edx
6844                            <1>       ; eax = byte count
6845 00002617 89C1              <1>       mov    ecx, eax
6846                            <1>       ;; 07/08/2016
6847                            <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
6848                            <1>       ;mov   ecx, edx
6849 00002619 88D8              <1>       mov    al, bl ; fill value for blanked lines
6850 0000261B BF00000A00        <1>       mov    edi, 0A0000h
6851 00002620 F3AA              <1>       rep    stosb
6852                            <1>
6853 00002622 66B80500          <1>       mov    ax, 5
6854 00002626 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6855 0000262A 66EF              <1>       out    dx, ax ; 0005h
6856                            <1>
6857 0000262C C3                <1>       retn
6858                            <1>
6859                            <1> vga_g_up_5:
6860                            <1>       ;;pop  ax
6861                            <1>       ; 12/04/2021
6862                            <1>       ;pop   eax
6863                            <1>
6864                            <1> vga_g_up_6:
6865                            <1>       ; [ESI] = VGA memory model number for current video mode
6866                            <1>         ;
6867                            <1>         ; LINEAR8 equ 5
6868                            <1>         ; PLANAR4 equ 4
6869                            <1>         ; PLANAR1 equ 3
6870                            <1>
6871 0000262D 803E04            <1>       cmp    byte [esi], PLANAR4
6872 00002630 7424              <1>       je     short vga_g_up_planar
6873 00002632 803E03            <1>       cmp    byte [esi], PLANAR1
6874 00002635 741F              <1>       je     short vga_g_up_planar
6875                            <1> vga_g_up_linear8:
6876                            <1>       ; 07/07/2016 (TEMPORARY)
6877                            <1>       ;
6878                            <1>       ; cl = upper left column ; cul
6879                            <1>       ; ch = upper left row ; rul
6880                            <1>       ; dl = lower rigth column ; clr
6881                            <1>       ; dh = lower right row ; rlr
6882                            <1>
6883                            <1> vga_g_up_l0:
```

```
6884                                    <1>       ;{for(i=rul;i<=rlr;i++)
6885                                    <1>       ; if((i+nblines>rlr)||(nblines==0))
6886 00002637 08C0                     <1>       or     al, al
6887 00002639 7414                     <1>       jz     short vga_g_up_l2
6888 0000263B 88C4                     <1>       mov    ah, al
6889 0000263D 00EC                     <1>       add    ah, ch ; i+nblines
6890                                    <1>       ;jc    short vga_g_up_l2
6891 0000263F 38F4                     <1>       cmp    ah, dh
6892 00002641 770C                     <1>       ja     short vga_g_up_l2
6893                                    <1>       ; else
6894                                    <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
6895 00002643 E8F2000000               <1>       call   vgamem_copy_l8
6896                                    <1> vga_g_up_l1:
6897 00002648 FEC5                     <1>       inc    ch
6898 0000264A 38F5                     <1>       cmp    ch, dh
6899 0000264C 76E9                     <1>       jna    short vga_g_up_l0
6900 0000264E C3                       <1>       retn
6901                                    <1> vga_g_up_l2:
6902                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
6903 0000264F E850010000               <1>       call   vgamem_fill_l8
6904 00002654 EBF2                     <1>       jmp    short vga_g_up_l1
6905                                    <1>
6906                                    <1> vga_g_up_planar:
6907                                    <1>       ; cl = upper left column ; cul
6908                                    <1>       ; ch = upper left row ; rul
6909                                    <1>       ; dl = lower rigth column ; clr
6910                                    <1>       ; dh = lower right row ; rlr
6911                                    <1> vga_g_up_pl0:
6912                                    <1>       ;{for(i=rul;i<=rlr;i++)
6913                                    <1>       ; if((i+nblines>rlr)||(nblines==0))
6914 00002656 20C0                     <1>       and    al, al
6915 00002658 7414                     <1>       jz     short vga_g_up_pl2
6916 0000265A 88C4                     <1>       mov    ah, al
6917 0000265C 00EC                     <1>       add    ah, ch ; i+nblines
6918                                    <1>       ;jc    short vga_g_up_pl2
6919 0000265E 38F4                     <1>       cmp    ah, dh
6920 00002660 770C                     <1>       ja     short vga_g_up_pl2
6921                                    <1>       ; else
6922                                    <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
6923 00002662 E80E000000               <1>       call   vgamem_copy_pl4
6924                                    <1> vga_g_up_pl1:
6925 00002667 FEC5                     <1>       inc    ch
6926 00002669 38F5                     <1>       cmp    ch, dh
6927 0000266B 76E9                     <1>       jna    short vga_g_up_pl0
6928 0000266D C3                       <1>       retn
6929                                    <1> vga_g_up_pl2:
6930                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
6931 0000266E E870000000               <1>       call   vgamem_fill_pl4
6932 00002673 EBF2                     <1>       jmp    short vga_g_up_pl1
6933                                    <1>
6934                                    <1> vgamem_copy_pl4:
6935                                    <1>       ; 08/08/2016
6936                                    <1>       ; 07/08/2016
6937                                    <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6938                                    <1>       ;
6939                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
6940                                    <1>       ; vgabios-0.7a (2011)
6941                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
6942                                    <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
6943                                    <1>       ;
6944                                    <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
6945                                    <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
6946                                    <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
6947                                    <1>
6948                                    <1>       ; src=ysrc*cheight*nbcols+xstart;
6949                                    <1>       ; dest=ydest*cheight*nbcols+xstart;
6950                                    <1>
6951 00002675 52                       <1>       push   edx
6952 00002676 50                       <1>       push   eax
6953                                    <1>
6954                                    <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
6955 00002677 66B80501                 <1>       mov    ax, 0105h
6956 0000267B 66BACE03                 <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6957 0000267F 66EF                     <1>       out    dx, ax
6958                                    <1>
6959                                    <1>       ; 07/08/2016
6960                                    <1>       ;mov     ah, [esp+1]
6961                                    <1>       ;movzx edx, ah ; ysrc
6962 00002681 0FB6542401               <1>       movzx  edx, byte [esp+1]
6963                                    <1>       ; 08/08/2016
6964 00002686 0FB605[9E6E0000]         <1>       movzx  eax, byte [CHAR_HEIGHT]
6965 0000268D 8A25[9C6E0000]           <1>       mov    ah, [CRT_COLS] ; nbcols
6966 00002693 F6E4                     <1>       mul    ah
6967                                    <1>       ;; 07/08/2016
6968                                    <1>       ;movzx eax, byte [CRT_COLS]
6969                                    <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
6970 00002695 50                       <1>       push   eax ; cheight * nbcols
6971 00002696 F7E2                     <1>       mul    edx ; * ysrc
6972                                    <1>       ; eax = ysrc * cheight * nbcols
6973                                    <1>       ; edx = 0
6974 00002698 88CA                     <1>       mov    dl, cl ; edx = xstart
6975 0000269A 01D0                     <1>       add    eax, edx
6976 0000269C 89C6                     <1>       mov    esi, eax ; src
6977 0000269E 88EA                     <1>       mov    dl, ch ; ydest
6978 000026A0 58                       <1>       pop    eax ; cheight * nbcols
6979 000026A1 F7E2                     <1>       mul    edx
6980                                    <1>       ; eax = ydest * cheight * nbcols
6981 000026A3 88CA                     <1>       mov    dl, cl ; edx = xstart
6982 000026A5 01D0                     <1>       add    eax, edx
6983 000026A7 89C7                     <1>       mov    edi, eax ; dest
6984                                    <1>       ; esi = src
6985                                    <1>       ; edi = dest
6986                                    <1>       ; for(i=0;i<cheight;i++)
6987                                    <1>       ; {
6988                                    <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
```

```
6989                                    <1>       ; }
6990 000026A9 51                        <1>       push   ecx
6991 000026AA B900000A00                <1>       mov    ecx, 0A0000h
6992 000026AF 01CE                      <1>       add    esi, ecx
6993 000026B1 01CF                      <1>       add    edi, ecx
6994                                    <1>       ; 08/08/2016
6995 000026B3 8A35[9E6E0000]            <1>       mov    dh, [CHAR_HEIGHT]
6996                                    <1>       ;; 07/08/2016
6997                                    <1>       ;mov   dh, 8 ; 07/08/2016
6998 000026B9 28D2                      <1>       sub    dl, dl ; i
6999                                    <1> vgamem_copy_pl4_0:
7000 000026BB 56                        <1>       push   esi
7001 000026BC 57                        <1>       push   edi
7002 000026BD 0FB605[9C6E0000]          <1>       movzx  eax, byte [CRT_COLS]
7003 000026C4 F6E2                      <1>       mul    dl
7004                                    <1>       ; eax = i * nbcols
7005 000026C6 01C7                      <1>       add    edi, eax ; dest+i*nbcols
7006 000026C8 01C6                      <1>       add    esi, eax
7007 000026CA 0FB6CF                    <1>       movzx  ecx, bh ; cols
7008 000026CD F3A4                      <1>       rep    movsb
7009 000026CF 5F                        <1>       pop    edi
7010 000026D0 5E                        <1>       pop    esi
7011 000026D1 FECE                      <1>       dec    dh
7012 000026D3 75E6                      <1>       jnz    short vgamem_copy_pl4_0
7013                                    <1> vgamem_copy_pl4_1:
7014 000026D5 59                        <1>       pop    ecx
7015                                    <1>
7016                                    <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7017 000026D6 66B80500                  <1>       mov    ax, 0005h
7018 000026DA 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7019 000026DE 66EF                      <1>       out    dx, ax
7020                                    <1>
7021 000026E0 58                        <1>       pop    eax
7022 000026E1 5A                        <1>       pop    edx
7023                                    <1>
7024 000026E2 C3                        <1>       retn
7025                                    <1>
7026                                    <1> vgamem_fill_pl4:
7027                                    <1>       ; 08/08/2016
7028                                    <1>       ; 07/08/2016
7029                                    <1>       ; 04/08/2016
7030                                    <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7031                                    <1>       ;
7032                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7033                                    <1>       ; vgabios-0.7a (2011)
7034                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7035                                    <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
7036                                    <1>       ;
7037                                    <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
7038                                    <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
7039                                    <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
7040                                    <1>
7041                                    <1>       ; dest=ystart*cheight*nbcols+xstart;
7042 000026E3 52                        <1>       push   edx
7043 000026E4 50                        <1>       push   eax
7044                                    <1>
7045                                    <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205)
7046 000026E5 66B80502                  <1>       mov    ax, 0205h
7047 000026E9 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7048 000026ED 66EF                      <1>       out    dx, ax
7049                                    <1>
7050                                    <1>              ; 08/08/2016
7051 000026EF 0FB605[9E6E0000]          <1>       movzx   eax, byte [CHAR_HEIGHT]
7052 000026F6 F6E5                      <1>       mul    ch
7053                                    <1>       ;; 07/08/2016
7054                                    <1>       ;movzx eax, ch
7055                                    <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
7056 000026F8 0FB615[9C6E0000]          <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
7057 000026FF F7E2                      <1>       mul    edx
7058                                    <1>       ; edx  = 0
7059 00002701 88CA                      <1>       mov    dl, cl
7060 00002703 01D0                      <1>       add    eax, edx
7061 00002705 89C7                      <1>       mov    edi, eax
7062                                    <1>       ; edi = dest
7063                                    <1>       ; for(i=0;i<cheight;i++)
7064                                    <1>       ; {
7065                                    <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
7066                                    <1>       ; }
7067 00002707 81C700000A00               <1>       add    edi, 0A0000h
7068 0000270D 51                        <1>       push   ecx
7069                                    <1>       ; 08/08/2016
7070 0000270E 8A35[9E6E0000]            <1>       mov    dh, [CHAR_HEIGHT]
7071                                    <1>       ;; 07/08/2016
7072                                    <1>       ;mov   dh, 8 ; 07/08/2016
7073 00002714 28D2                      <1>       sub    dl, dl ; i
7074                                    <1> vgamem_fill_pl4_0:
7075 00002716 57                        <1>       push   edi
7076 00002717 0FB605[9C6E0000]          <1>       movzx  eax, byte [CRT_COLS]
7077 0000271E F6E2                      <1>       mul    dl
7078                                    <1>       ; eax = i * nbcols
7079 00002720 01C7                      <1>       add    edi, eax ; dest+i*nbcols
7080 00002722 88D8                      <1>       mov    al, bl ; attr ; 04/08/2016
7081 00002724 0FB6CF                    <1>       movzx  ecx, bh ; cols
7082 00002727 F3AA                      <1>       rep    stosb
7083 00002729 5F                        <1>       pop    edi
7084 0000272A 75EA                      <1>       jnz    short vgamem_fill_pl4_0
7085                                    <1> vgamem_fill_pl4_1:
7086 0000272C 59                        <1>       pop    ecx
7087                                    <1>
7088                                    <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7089 0000272D 66B80500                  <1>       mov    ax, 0005h
7090 00002731 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7091 00002735 66EF                      <1>       out    dx, ax
7092                                    <1>
7093 00002737 58                        <1>       pop    eax
```

```
7094 00002738 5A                <1>       pop    edx
7095                            <1>
7096 00002739 C3                <1>       retn
7097                            <1>
7098                            <1> vgamem_copy_l8:
7099                            <1>       ; 08/08/2016
7100                            <1>       ; 07/08/2016
7101                            <1>       ; 06/08/2016
7102                            <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7103                            <1>       ;
7104                            <1>       ; TEMPORARY
7105                            <1>       ;
7106                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7107                            <1>       ; vgabios-0.7a (2011)
7108                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7109                            <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
7110                            <1>       ;
7111                            <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
7112                            <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
7113                            <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
7114                            <1>
7115                            <1>       ; src=ysrc*cheight*nbcols+xstart;
7116                            <1>       ; dest=ydest*cheight*nbcols+xstart;
7117                            <1>
7118 0000273A 52               <1>       push   edx
7119 0000273B 50               <1>       push   eax
7120                            <1>
7121                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
7122                            <1>       ;mov   ax, 0105h
7123                            <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7124                            <1>       ;out   dx, ax
7125                            <1>
7126                            <1>       ;mov   ah, [esp+1]
7127                            <1>
7128 0000273C 0FB6D4           <1>       movzx  edx, ah ; ysrc
7129                            <1>       ; 08/08/2016
7130 0000273F 0FB605[9E6E0000]  <1>       movzx  eax, byte [CHAR_HEIGHT]
7131 00002746 8A25[9C6E0000]    <1>       mov    ah, [CRT_COLS] ; nbcols
7132 0000274C F6E4              <1>       mul    ah
7133                            <1>       ;; 07/08/2016
7134                            <1>       ;movzx eax, byte [CRT_COLS]
7135                            <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
7136 0000274E 50                <1>       push   eax ; cheight * nbcols
7137 0000274F F7E2              <1>       mul    edx ; * ysrc
7138                            <1>       ; eax = ysrc * cheight * nbcols
7139                            <1>       ; edx = 0
7140 00002751 88CA              <1>       mov    dl, cl ; edx = xstart
7141 00002753 01D0              <1>       add    eax, edx
7142 00002755 89C6              <1>       mov    esi, eax ; src
7143 00002757 66C1E603          <1>       shl    si, 3 ; * 8 ; 06/08/2016
7144 0000275B 88EA              <1>       mov    dl, ch ; ydest
7145 0000275D 58                <1>       pop    eax ; cheight * nbcols
7146 0000275E F7E2              <1>       mul    edx
7147                            <1>       ; eax = ydest * cheight * nbcols
7148 00002760 88CA              <1>       mov    dl, cl ; edx = xstart
7149 00002762 01D0              <1>       add    eax, edx
7150 00002764 89C7              <1>       mov    edi, eax ; dest
7151 00002766 66C1E703          <1>       shl    di, 3 ; * 8 ; 06/08/2016
7152                            <1>       ; esi = src
7153                            <1>       ; edi = dest
7154                            <1>       ; for(i=0;i<cheight;i++)
7155                            <1>       ; {
7156                            <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
7157                            <1>       ; }
7158 0000276A 51                <1>       push   ecx
7159 0000276B B900000A00        <1>       mov    ecx, 0A0000h
7160 00002770 01CE              <1>       add    esi, ecx
7161 00002772 01CF              <1>       add    edi, ecx
7162                            <1>       ; 08/08/2016
7163 00002774 8A35[9E6E0000]    <1>       mov    dh, [CHAR_HEIGHT]
7164                            <1>       ;; 07/08/2016
7165                            <1>       ;mov   dh, 8 ; 07/08/2016
7166 0000277A 28D2              <1>       sub    dl, dl ; i
7167                            <1> vgamem_copy_l8_0:
7168 0000277C 56                <1>       push   esi
7169 0000277D 57                <1>       push   edi
7170 0000277E 0FB605[9C6E0000]  <1>       movzx  eax, byte [CRT_COLS]
7171 00002785 F6E2              <1>       mul    dl
7172                            <1>       ; eax = i * nbcols
7173 00002787 66C1E003          <1>       shl    ax, 3 ; * 8 ; 06/08/2016
7174 0000278B 01C7              <1>       add    edi, eax ; dest+i*nbcols
7175 0000278D 01C6              <1>       add    esi, eax
7176 0000278F 0FB6CF            <1>       movzx  ecx, bh ; cols
7177 00002792 66C1E103          <1>       shl    cx, 3 ; * 8 ; 06/08/2016
7178 00002796 F3A4              <1>       rep    movsb
7179 00002798 5F                <1>       pop    edi
7180 00002799 5E                <1>       pop    esi
7181 0000279A FEC2              <1>       inc    dl ; 06/08/2016
7182 0000279C FECE              <1>       dec    dh
7183 0000279E 75DC              <1>       jnz    short vgamem_copy_l8_0
7184                            <1> vgamem_copy_l8_1:
7185 000027A0 59                <1>       pop    ecx
7186                            <1>
7187                            <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7188                            <1>       ;mov   ax, 0005h
7189                            <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7190                            <1>       ;out   dx, ax
7191                            <1>
7192 000027A1 58                <1>       pop    eax
7193 000027A2 5A                <1>       pop    edx
7194                            <1>
7195 000027A3 C3                <1>       retn
7196                            <1>
7197                            <1> vgamem_fill_l8:
7198                            <1>       ; 08/08/2016
```

```
7199                                    <1>       ; 07/08/2016
7200                                    <1>       ; 06/08/2016
7201                                    <1>       ; 04/08/2016
7202                                    <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7203                                    <1>       ;
7204                                    <1>       ; TEMPORARY
7205                                    <1>       ;
7206                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7207                                    <1>       ; vgabios-0.7a (2011)
7208                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7209                                    <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
7210                                    <1>       ;
7211                                    <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
7212                                    <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
7213                                    <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
7214                                    <1>
7215                                    <1>       ; dest=ystart*cheight*nbcols+xstart;
7216 000027A4 52                       <1>       push   edx
7217 000027A5 50                       <1>       push   eax
7218                                    <1>
7219                                    <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0205)
7220                                    <1>       ;mov   ax, 0205h
7221                                    <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7222                                    <1>       ;out   dx, ax
7223                                    <1>
7224                                    <1>         ; 08/08/2016
7225 000027A6 0FB605[9E6E0000]         <1>       movzx   eax, byte [CHAR_HEIGHT]
7226 000027AD F6E5                     <1>       mul    ch
7227                                    <1>       ;; 07/08/2016
7228                                    <1>       ;movzx eax, ch
7229                                    <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
7230 000027AF 0FB615[9C6E0000]         <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
7231 000027B6 F7E2                     <1>       mul    edx
7232                                    <1>       ; edx  = 0
7233 000027B8 88CA                     <1>       mov    dl, cl
7234 000027BA 01D0                     <1>       add    eax, edx
7235 000027BC 89C7                     <1>       mov    edi, eax
7236 000027BE 66C1E703                 <1>       shl    di, 3 ; * 8 ; 06/08/2016
7237                                    <1>       ; edi = dest
7238                                    <1>       ; for(i=0;i<cheight;i++)
7239                                    <1>       ; {
7240                                    <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
7241                                    <1>       ; }
7242 000027C2 81C700000A00               <1>       add    edi, 0A0000h
7243 000027C8 51                       <1>       push   ecx
7244                                    <1>       ; 08/08/2016
7245 000027C9 8A35[9E6E0000]           <1>       mov    dh, [CHAR_HEIGHT]
7246                                    <1>       ;; 07/08/2016
7247                                    <1>       ;mov   dh, 8 ; 07/08/2016
7248 000027CF 28D2                     <1>       sub    dl, dl ; i
7249                                    <1> vgamem_fill_l8_0:
7250 000027D1 57                       <1>       push   edi
7251 000027D2 0FB605[9C6E0000]         <1>       movzx  eax, byte [CRT_COLS]
7252 000027D9 F6E2                     <1>       mul    dl
7253                                    <1>       ; eax = i * nbcols
7254 000027DB 66C1E003                 <1>       shl    ax, 3 ; * 8 ; 06/08/2016
7255 000027DF 01C7                     <1>       add    edi, eax ; dest+i*nbcols
7256 000027E1 88D8                     <1>       mov    al, bl ; attr ; 04/08/2016
7257 000027E3 0FB6CF                   <1>       movzx  ecx, bh ; cols
7258 000027E6 66C1E103                 <1>       shl    cx, 3 ; * 8 ; 06/08/2016
7259 000027EA F3AA                     <1>       rep    stosb
7260 000027EC 5F                       <1>       pop    edi
7261 000027ED FEC2                     <1>       inc    dl ; 06/08/2016
7262 000027EF FECE                     <1>       dec    dh
7263 000027F1 75DE                     <1>       jnz    short vgamem_fill_l8_0
7264                                    <1> vgamem_fill_l8_1:
7265 000027F3 59                       <1>       pop    ecx
7266                                    <1>
7267                                    <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7268                                    <1>       ;mov   ax, 0005h
7269                                    <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7270                                    <1>       ;out   dx, ax
7271                                    <1>
7272 000027F4 58                       <1>       pop    eax
7273 000027F5 5A                       <1>       pop    edx
7274                                    <1>
7275 000027F6 C3                       <1>       retn
7276                                    <1>
7277                                    <1> vga_graphics_down:
7278                                    <1>       ; 12/04/2021
7279                                    <1>       ; 08/08/2016
7280                                    <1>       ; 07/08/2016
7281                                    <1>       ; 31/07/2016
7282                                    <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7283                                    <1>       ;
7284                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7285                                    <1>       ; vgabios-0.7a (2011)
7286                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7287                                    <1>       ; 'vgabios.c', 'biosfn_scroll'
7288                                    <1>       ;
7289                                    <1>
7290                                    <1>       ; cl = upper left column
7291                                    <1>       ; ch = upper left row
7292                                    <1>       ; dl = lower rigth column
7293                                    <1>       ; dh = lower right row
7294                                    <1>       ;
7295                                    <1>       ; al = line count (AL=0 means blank entire fields)
7296                                    <1>       ; bl = fill value for blanked lines
7297                                    <1>       ; bh = unused
7298                                    <1>       ;
7299                                    <1>       ; ah = [CRT_MODE], current video mode
7300                                    <1>
7301 000027F7 FC                       <1>       cld     ; !!! Clear direction flag !!!
7302                                    <1>
7303 000027F8 88C7                     <1>       mov    bh, al ; 31/07/2016
```

```
7304                                    <1>
7305 000027FA BE[B66E0000]              <1>       mov    esi, vga_modes
7306 000027FF 89F7                      <1>       mov    edi, esi
7307 00002801 83C710                    <1>       add    edi, vga_mode_count
7308                                    <1> vga_g_down_0:
7309 00002804 AC                        <1>       lodsb
7310 00002805 38E0                      <1>       cmp    al, ah ; [CRT_MODE]
7311 00002807 7405                      <1>       je     short vga_g_down_1
7312 00002809 39FE                      <1>       cmp    esi, edi
7313 0000280B 72F7                      <1>       jb     short vga_g_down_0
7314                                    <1>       ; xor  bh, bh ; 31/07/2016
7315 0000280D C3                        <1>       retn   ; nothing to do
7316                                    <1> vga_g_down_1:
7317 0000280E 88F8                      <1>       mov    al, bh ; 31/07/2016
7318 00002810 83C64F                    <1>       add    esi, vga_memmodel - (vga_modes + 1)
7319                                    <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
7320                                    <1>
7321                                    <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
7322                                    <1>       ; if(clr>=nbcols)clr=nbcols-1;
7323                                    <1>       ; if(nblines>nbrows)nblines=0;
7324                                    <1>       ; cols=clr-cul+1;
7325                                    <1>
7326 00002813 3A35[A26E0000]            <1>       cmp    dh, [VGA_ROWS]
7327 00002819 7208                      <1>       jb     short vga_g_down_2
7328 0000281B 8A35[A26E0000]            <1>       mov    dh, [VGA_ROWS]
7329 00002821 FECE                      <1>       dec    dh
7330                                    <1> vga_g_down_2:
7331 00002823 3A15[9C6E0000]            <1>       cmp    dl, [CRT_COLS]  ; = [VGA_COLS]
7332 00002829 7208                      <1>       jb     short vga_g_down_3
7333 0000282B 8A15[9C6E0000]            <1>       mov    dl, [CRT_COLS]
7334 00002831 FECA                      <1>       dec    dl
7335                                    <1> vga_g_down_3:
7336 00002833 3A05[A26E0000]            <1>       cmp    al, [VGA_ROWS]
7337 00002839 7602                      <1>       jna    short vga_g_down_4
7338 0000283B 28C0                      <1>       sub    al, al ; 0
7339                                    <1> vga_g_down_4:
7340 0000283D 88F7                      <1>       mov    bh, dh ; clr
7341 0000283F 28CF                      <1>       sub    bh, cl ; cul
7342 00002841 FEC7                      <1>       inc    bh ; cols = clr-cul+1
7343                                    <1>
7344 00002843 20C0                      <1>       and    al, al ; nblines = 0
7345 00002845 7557                      <1>       jnz    short vga_g_down_6
7346 00002847 20ED                      <1>       and    ch, ch ; rul = 0
7347 00002849 7553                      <1>       jnz    short vga_g_down_6
7348 0000284B 20C9                      <1>       and    cl, cl ; cul = 0
7349 0000284D 754F                      <1>       jnz    short vga_g_down_6
7350                                    <1>
7351 0000284F 50                        <1>       push   eax ; push ax ; 12/04/2021
7352 00002850 A0[A26E0000]              <1>       mov    al, [VGA_ROWS]
7353 00002855 FEC8                      <1>       dec    al
7354 00002857 38C6                      <1>       cmp    dh, al ; rlr = nbrows-1
7355 00002859 7543                      <1>       jne    short vga_g_down_5
7356 0000285B A0[9C6E0000]              <1>       mov   al, [CRT_COLS]  ; = VGA_COLS
7357 00002860 FEC8                      <1>       dec    al
7358 00002862 38C2                      <1>       cmp    dl, al ; clr = nbcols-1
7359                                    <1>       ;jne   short vga_g_down_5
7360                                    <1>       ; 12/04/2021
7361 00002864 58                        <1>       pop    eax ; pop ax
7362 00002865 7537                      <1>       jne    short vga_g_down_5
7363                                    <1>
7364 00002867 66B80502                  <1>       mov    ax, 0205h
7365 0000286B 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7366 0000286F 66EF                      <1>       out    dx, ax
7367 00002871 A0[A26E0000]              <1>       mov    al, [VGA_ROWS]
7368 00002876 8A25[9C6E0000]            <1>       mov    ah, [CRT_COLS] ; = [VGA_COLS]
7369 0000287C F6E4                      <1>       mul    ah
7370 0000287E 0FB7D0                    <1>       movzx  edx, ax
7371                                    <1>       ; 08/08/2016
7372 00002881 0FB605[9E6E0000]          <1>       movzx  eax, byte [CHAR_HEIGHT]
7373 00002888 F7E2                      <1>       mul    edx
7374                                    <1>       ; eax = byte count
7375 0000288A 89C1                      <1>       mov    ecx, eax
7376                                    <1>       ;; 07/08/2016
7377                                    <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
7378                                    <1>       ;mov   ecx, edx
7379 0000288C 88D8                      <1>       mov    al, bl ; fill value for blanked lines
7380 0000288E BF00000A00                <1>       mov    edi, 0A0000h
7381 00002893 F3AA                      <1>       rep    stosb
7382                                    <1>
7383 00002895 B005                      <1>       mov    al, 5
7384 00002897 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7385 0000289B 66EF                      <1>       out    dx, ax ; 0005h
7386                                    <1>
7387 0000289D C3                        <1>       retn
7388                                    <1>
7389                                    <1> vga_g_down_5:
7390                                    <1>       ; 12/04/2021
7391                                    <1>       ;pop   eax  ; pop ax
7392                                    <1>
7393                                    <1> vga_g_down_6:
7394                                    <1>       ; [ESI] = VGA memory model number for current video mode
7395                                    <1>         ;
7396                                    <1>         ; LINEAR8 equ 5
7397                                    <1>         ; PLANAR4 equ 4
7398                                    <1>         ; PLANAR1 equ 3
7399                                    <1>
7400 0000289E 803E04                    <1>       cmp    byte [esi], PLANAR4
7401 000028A1 742C                      <1>       je     short vga_g_down_planar
7402 000028A3 803E03                    <1>       cmp    byte [esi], PLANAR1
7403 000028A6 7427                      <1>       je     short vga_g_down_planar
7404                                    <1> vga_g_down_linear8:
7405                                    <1>       ; 07/07/2016 (TEMPORARY)
7406                                    <1>       ;
7407                                    <1>       ; cl = upper left column ; cul
7408                                    <1>       ; ch = upper left row ; rul
```

```
                                <1>       ; dl = lower rigth column ; clr
                                <1>       ; dh = lower right row ; rlr
                                <1>
                                <1> vga_g_down_l0:
                                <1>       ;{for(i=rlr;i>=rul;i--)
                                <1>         ; if((i<rul+nblines)||(nblines==0))
000028A8 08C0                   <1>       or     al, al
000028AA 741C                   <1>       jz     short vga_g_down_l2
000028AC 88C4                   <1>       mov    ah, al
000028AE 00EC                   <1>       add    ah, ch
                                <1>       ;jc    short vga_g_down_l2
000028B0 86EE                   <1>       xchg   ch, dh
000028B2 38E5                   <1>       cmp    ch, ah
000028B4 7212                   <1>       jb     short vga_g_down_l2
000028B6 88EC                   <1>       mov    ah, ch
000028B8 28C4                   <1>       sub    ah, al ; ah = i - nblines
                                <1>       ; else
                                <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
000028BA E87BFEFFFF             <1>       call   vgamem_copy_l8
                                <1> vga_g_down_l1:
000028BF 86F5                   <1>       xchg   dh, ch
000028C1 FECE                   <1>       dec    dh
000028C3 38EE                   <1>       cmp    dh, ch
000028C5 73E1                   <1>       jnb    short vga_g_down_l0
000028C7 C3                     <1>       retn
                                <1>
                                <1> vga_g_down_l2:
                                <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
000028C8 E8D7FEFFFF             <1>       call   vgamem_fill_l8
000028CD EBF0                   <1>       jmp    short vga_g_down_l1
                                <1>
                                <1> vga_g_down_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_down_pl0:
                                <1>       ;{for(i=rlr;i>=rul;i--)
                                <1>         ; if((i<rul+nblines)||(nblines==0))
000028CF 08C0                   <1>       or     al, al
000028D1 741C                   <1>       jz     short vga_g_down_pl2
000028D3 88C4                   <1>       mov    ah, al
000028D5 00EC                   <1>       add    ah, ch
                                <1>       ;jc    short vga_g_down_pl2
000028D7 86EE                   <1>       xchg   ch, dh
000028D9 38E5                   <1>       cmp    ch, ah
000028DB 7212                   <1>       jb     short vga_g_down_pl2
000028DD 88EC                   <1>       mov    ah, ch
000028DF 28C4                   <1>       sub    ah, al ; ah = i - nblines
                                <1>       ; else
                                <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
000028E1 E88FFDFFFF             <1>       call   vgamem_copy_pl4
                                <1> vga_g_down_pl1:
000028E6 86F5                   <1>       xchg   dh, ch
000028E8 FECE                   <1>       dec    dh
000028EA 38EE                   <1>       cmp    dh, ch
000028EC 73E1                   <1>       jnb    short vga_g_down_pl0
000028EE C3                     <1>       retn
                                <1>
                                <1> vga_g_down_pl2:
                                <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
000028EF E8EFFDFFFF             <1>       call   vgamem_fill_pl4
000028F4 EBF0                   <1>       jmp    short vga_g_down_pl1
                                <1>
                                <1> ; 07/07/2016
                                <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;--------------------------------------------------
                                <1> ; SCROLL DOWN
                                <1> ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
                                <1> ; ENTRY --
                                <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
                                <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
                                <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
                                <1> ;  BH = FILL VALUE FOR BLANKED LINES
                                <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
                                <1> ;  DS = DATA SEGMENT
                                <1> ;  ES = REGEN SEGMENT
                                <1> ; EXIT --
                                <1> ;  NOTHING, THE SCREEN IS SCROLLED
                                <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> GRAPHICS_DOWN:
                                <1>       ; 07/07/2016
                                <1>       ;AH = Current video mode, [CRT_MODE]
                                <1>       ;STD                    ; SET DIRECTION
000028F6 80FC07                 <1>       cmp    ah, 7
000028F9 0F87F8FEFFFF           <1>       ja    vga_graphics_down
                                <1>       ;je    _n0
                                <1>
000028FF 88C7                   <1>       MOV    bh, al           ; save line count in BH
00002901 6689D0                 <1>       MOV    AX, DX           ; GET LOWER RIGHT POSITION INTO AX REG
                                <1>
                                <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
                                <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                                <1>
```

```
7514 00002904 E8F0010000          <1>       CALL   GRAPH_POSN
7515 00002909 0FB7F8              <1>       MOVzx  eDI, AX             ; SAVE RESULT AS DESTINATION ADDRESS
7516                              <1>
7517                              <1> ;-----      DETERMINE SIZE OF WINDOW
7518                              <1>
7519 0000290C 6629CA              <1>       SUB    DX, CX
7520 0000290F 6681C20101          <1>       ADD    DX, 101h            ; ADJUST VALUES
7521 00002914 C0E602              <1>       SAL    DH, 2               ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
7522                              <1>                                  ; AND EVEN/ODD ROWS
7523                              <1>
7524                              <1> ;-----      DETERMINE CRT MODE
7525                              <1>
7526 00002917 803D[9A6E0000]06    <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
7527 0000291E 7307                <1>       JNC    short _R12          ; FIND_SOURCE_DOWN
7528                              <1>
7529                              <1> ;-----      MEDIUM RES DOWN
7530 00002920 D0E2                <1>       SAL    DL, 1               ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
7531 00002922 66D1E7              <1>       SAL    DI, 1               ; OFFSET *2 SINCE 2 BYTES/CHAR
7532 00002925 6647                <1>       INC    DI                  ; POINT TO LAST BYTE
7533                              <1>
7534                              <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
7535                              <1>
7536                              <1> _R12:                           ; FIND_SOURCE_DOWN
7537 00002927 81C700800B00        <1>       add    edi, 0B8000h
7538 0000292D 6681C7F000          <1>       ADD    DI, 240             ; POINT TO LAST ROW OF PIXELS
7539 00002932 C0E702              <1>       sal    bh, 2               ; multiply number of lines by 4
7540 00002935 74(06)              <1>       JZ     short 6             ; IF ZERO, THEN BLANK ENTIRE FIELD
7541 00002937 B050                <1>       MOV    AL, 80              ; 80 BYTES/ROW
7542 00002939 F6E7                <1>       mul    bh                  ; determine offset to source
7543 0000293B 89FE                <1>       MOV    eSI, eDI            ; SET UP SOURCE
7544 0000293D 6629C6              <1>       SUB    SI, AX              ; SUBTRACT THE OFFSET
7545 00002940 88F4                <1>       MOV    AH, DH              ; NUMBER OF ROWS IN FIELD
7546 00002942 28FC                <1>       sub    ah, bh              ; determine number to move
7547                              <1>
7548                              <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
7549                              <1>
7550                              <1> _R13:                           ; ROW_LOOP_DOWN
7551 00002944 E824000000          <1>       CALL   _R17                ; MOVE ONE ROW
7552 00002949 6681EE5020          <1>       SUB    SI, 2000h+80        ; MOVE TO NEXT ROW
7553 0000294E 6681EF5020          <1>       SUB    DI, 2000h+80
7554 00002953 FECC                <1>       DEC    AH                  ; NUMBER OF ROWS TO MOVE
7555 00002955 75ED                <1>       JNZ    short _R13          ; CONTINUE TILL ALL MOVED
7556                              <1>
7557                              <1> ;-----      FILL IN THE VACATED LINE(S)
7558                              <1> _R14:                           ; CLEAR_ENTRY_DOWN
7559 00002957 88D8                <1>       mov    al, bl              ; attribute to fill with
7560                              <1> _R15_:                          ; CLEAR_LOOP_DOWN
7561 00002959 E82B000000          <1>       CALL   _R18                ; CLEAR A ROW
7562 0000295E 6681EF5020          <1>       SUB    DI, 2000h+80        ; POINT TO NEXT LINE
7563 00002963 FECF                <1>       dec    bh                  ; number of lines to fill
7564 00002965 75F2                <1>       JNZ    short _R15_         ; CLEAR_LOOP_DOWN
7565                              <1>       ; 18/11/2020
7566 00002967 FC                  <1>       CLD                        ; RESET THE DIRECTION FLAG
7567                              <1>
7568 00002968 C3                  <1>       retn                       ; EVERYTHING DONE
7569                              <1>
7570                              <1> _R16:                           ; BLANK_FIELD_DOWN
7571 00002969 88F7                <1>       mov    bh, dh              ; set blank count to everything in field
7572 0000296B EBEA                <1>       JMP    short _R14          ; CLEAR THE FIELD
7573                              <1>
7574                              <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
7575                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7576                              <1>
7577                              <1> ;-----      ROUTINE TO MOVE ONE ROW OF INFORMATION
7578                              <1>
7579                              <1> _R17:
7580 0000296D 0FB6CA              <1>       MOVzx  ecx, DL             ; NUMBER OF BYTES IN THE ROW
7581 00002970 56                  <1>       PUSH   eSI
7582 00002971 57                  <1>       PUSH   eDI                 ; SAVE POINTERS
7583 00002972 F3A4                <1>       REP    MOVSB               ; MOVE THE EVEN FIELD
7584 00002974 5F                  <1>       POP    eDI
7585 00002975 5E                  <1>       POP    eSI
7586 00002976 6681C60020          <1>       ADD    SI, 2000h
7587 0000297B 6681C70020          <1>       ADD    DI, 2000h           ; POINT TO THE ODD FIELD
7588 00002980 56                  <1>       PUSH   eSI
7589 00002981 57                  <1>       PUSH   eDI                 ; SAVE THE POINTERS
7590 00002982 88D1                <1>       MOV    CL, DL              ; COUNT BACK
7591 00002984 F3A4                <1>       REP    MOVSB               ; MOVE THE ODD FIELD
7592 00002986 5F                  <1>       POP    eDI
7593 00002987 5E                  <1>       POP    eSI                 ; POINTERS BACK
7594 00002988 C3                  <1>       RETn                       ; RETURN TO CALLER
7595                              <1>
7596                              <1> ;-----      CLEAR A SINGLE ROW
7597                              <1>
7598                              <1> _R18:
7599 00002989 0FB6CA              <1>       MOVzx  ecx, DL             ; NUMBER OF BYTES IN FIELD
7600 0000298C 57                  <1>       PUSH   eDI                 ; SAVE POINTER
7601 0000298D F3AA                <1>       REP    STOSB               ; STORE THE NEW VALUE
7602 0000298F 5F                  <1>       POP    eDI                 ; POINTER BACK
7603 00002990 6681C70020          <1>       ADD    DI, 2000h           ; POINT TO ODD FIELD
7604 00002995 57                  <1>       PUSH   eDI
7605 00002996 88D1                <1>       MOV    CL, DL
7606 00002998 F3AA                <1>       REP    STOSB               ; FILL THE ODD FIELD
7607 0000299A 5F                  <1>       POP    eDI
7608 0000299B C3                  <1>       RETn                       ; RETURN TO CALLER
7609                              <1>
7610                              <1> ; 04/07/2016
7611                              <1> ; 01/07/2016
7612                              <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7613                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7614                              <1> ;-----------------------------------------------
7615                              <1> ; GRAPHICS WRITE
7616                              <1> ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
7617                              <1> ;  POSITION ON THE SCREEN.
7618                              <1> ; ENTRY --
```

```
                        <1> ;  AL = CHARACTER TO WRITE
                        <1> ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
                        <1> ;     IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
                        <1> ;     (0 IS USED FOR THE BACKGROUND COLOR)
                        <1> ;  CX = NUMBER OF CHARS TO WRITE
                        <1> ;  DS = DATA SEGMENT
                        <1> ;  ES = REGEN SEGMENT
                        <1> ; EXIT --
                        <1> ;  NOTHING IS RETURNED
                        <1> ;
                        <1> ; GRAPHICS READ
                        <1> ;  THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT CURSOR
                        <1> ;  POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
                        <1> ;  CHARACTER GENERATOR CODE POINTS
                        <1> ; ENTRY --
                        <1> ;  NONE (0 IS ASSUMED AS THE BACKGROUND COLOR)
                        <1> ; EXIT --
                        <1> ;  AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF NONE FOUND)
                        <1> ;
                        <1> ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE CONTAINED IN ROM
                        <1> ;  FOR THE 1ST 128 CHARS.  TO ACCESS CHARS IN THE SECOND HALF, THE USER
                        <1> ;  MUST INITIALIZE THE VECTOR AT INTERRUPT 1FH (LOCATION 0007CH) TO
                        <1> ;  POINT TO THE USER SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).
                        <1> ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS
                        <1> ;--------------------------------------------------
                        <1>
                        <1> GRAPHICS_WRITE:
0000299C 25FF000000     <1>       and   eax, 0FFh           ; ZERO TO HIGH OF CODE POINT
000029A1 50             <1>       PUSH  eAX                 ; SAVE CODE POINT VALUE
                        <1>
                        <1> ;-----      DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
                        <1>
000029A2 E84B010000     <1>       CALL  S26                 ; FIND LOCATION IN REGEN BUFFER
000029A7 89C7           <1>       MOV   eDI, eAX            ; REGEN POINTER IN DI
                        <1>
                        <1> ;-----      DETERMINE REGION TO GET CODE POINTS FROM
                        <1>
000029A9 58             <1>       POP   eAX                 ; RECOVER CODE POINT
                        <1>
000029AA BE[845F0100]   <1>       MOV   eSI, CRT_CHAR_GEN   ; OFFSET OF IMAGES
                        <1>
                        <1> ;-----      DETERMINE GRAPHICS MODE IN OPERATION
                        <1>                                 ; DETERMINE_MODE
000029AF 66C1E003       <1>       SAL   AX, 3               ; MULTIPLY CODE POINT VALUE BY 8
000029B3 01C6           <1>       ADD   eSI, eAX            ; SI HAS OFFSET OF DESIRED CODES
                        <1>
000029B5 803D[9A6E0000]06 <1>     CMP   byte [CRT_MODE], 6
000029BC 7231           <1>       JC    short S6            ; TEST FOR MEDIUM RESOLUTION MODE
                        <1>
                        <1> ;-----      HIGH RESOLUTION MODE
                        <1>
000029BE 81C700800B00   <1>       add   edi, 0B8000h
                        <1> S1:                             ; HIGH_CHAR
000029C4 57             <1>       PUSH  eDI                 ; SAVE REGEN POINTER
000029C5 56             <1>       PUSH  eSI                 ; SAVE CODE POINTER
000029C6 B604           <1>       MOV   DH, 4               ; NUMBER OF TIMES THROUGH LOOP
                        <1> S2:
000029C8 AC             <1>       LODSB                     ; GET BYTE FROM CODE POINTS
000029C9 F6C380         <1>       TEST  BL, 80H                   ; SHOULD WE USE THE FUNCTION
000029CC 7515           <1>       JNZ   short S5            ; TO PUT CHAR IN
000029CE AA             <1>       STOSB                     ; STORE IN REGEN BUFFER
000029CF AC             <1>       LODSB
                        <1> S4:
000029D0 8887FF1F0000   <1>       MOV   [eDI+2000H-1], AL ; STORE IN SECOND HALF
000029D6 83C74F         <1>       ADD   eDI, 79                   ; MOVE TO NEXT ROW IN REGEN
000029D9 FECE           <1>       DEC   DH                  ; DONE WITH LOOP
000029DB 75EB           <1>       JNZ   short S2
000029DD 5E             <1>       POP   eSI
000029DE 5F             <1>       POP   eDI                 ; RECOVER REGEN POINTER
000029DF 47             <1>       INC   eDI                 ; POINT TO NEXT CHAR POSITION
000029E0 E2E2           <1>       LOOP  S1                  ; MORE CHARS TO WRITE
000029E2 C3             <1>       retn
                        <1>
                        <1> S5:
000029E3 3207           <1>       XOR   AL, [eDI]           ; EXCLUSIVE OR WITH CURRENT
000029E5 AA             <1>       STOSB                     ; STORE THE CODE POINT
000029E6 AC             <1>       LODSB                     ; AGAIN FOR ODD FIELD
000029E7 3287FF1F0000   <1>       XOR   AL, [eDI+2000H-1]
000029ED EBE1           <1>       JMP   short S4            ; BACK TO MAINSTREAM
                        <1>
                        <1> ;-----      MEDIUM RESOLUTION WRITE
                        <1> S6:                             ; MED_RES_WRITE
000029EF 88DA           <1>       MOV   DL, BL              ; SAVE HIGH COLOR BIT
000029F1 66D1E7         <1>       SAL   DI, 1               ; OFFSET*2 SINCE 2 BYTES/CHAR
                        <1>                                 ; EXPAND BL TO FULL WORD OF COLOR
000029F4 80E303         <1>       AND   BL, 3               ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
000029F7 B055           <1>       MOV   AL, 055H            ; GET BIT CONVERSION MULTIPLIER
000029F9 F6E3           <1>       MUL   BL                  ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
000029FB 88C3           <1>       MOV   BL, AL              ; PLACE BACK IN WORK REGISTER
000029FD 88C7           <1>       MOV   BH, AL              ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
000029FF 81C700800B00   <1>       add   edi, 0B8000h
                        <1> S7:                                   ; MED_CHAR
00002A05 57             <1>       PUSH  eDI                 ; SAVE REGEN POINTER
00002A06 56             <1>       PUSH  eSI                 ; SAVE THE CODE POINTER
00002A07 B604           <1>       MOV   DH, 4               ; NUMBER OF LOOPS
                        <1> S8:
00002A09 AC             <1>       LODSB                     ; GET CODE POINT
00002A0A E8B3000000     <1>       CALL  S21                 ; DOUBLE UP ALL THE BITS
00002A0F 6621D8         <1>       AND   AX, BX              ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
00002A12 86E0           <1>       XCHG  AH, AL              ; SWAP HIGH/LOW BYTES FOR WORD MOVE
00002A14 F6C280         <1>       TEST  DL, 80H                   ; IS THIS XOR FUNCTION
00002A17 7403           <1>       JZ    short S9            ; NO, STORE IT IN AS IS
00002A19 663307         <1>       XOR   AX, [eDI]           ; DO FUNCTION WITH LOW/HIGH
                        <1> S9:
00002A1C 668907         <1>       MOV   [eDI], AX           ; STORE FIRST BYTE HIGH, SECOND LOW
```

```
7724 00002A1F AC                <1>       LODSB                   ; GET CODE POINT
7725 00002A20 E89D000000        <1>       CALL   S21
7726 00002A25 6621D8            <1>       AND    AX, BX           ; CONVERT TO COLOR
7727 00002A28 86E0              <1>       XCHG   AH, AL           ; SWAP HIGH/LOW BYTES FOR WORD MOVE
7728 00002A2A F6C280            <1>       TEST   DL, 80H                ; AGAIN, IS THIS XOR FUNCTION
7729 00002A2D 7407              <1>       JZ     short _S10       ; NO, JUST STORE THE VALUES
7730 00002A2F 66338700200000    <1>       XOR    AX, [eDI+2000H]         ; FUNCTION WITH FIRST HALF LOW
7731                            <1> _S10:
7732 00002A36 66898700200000    <1>       MOV    [eDI+2000H], AX         ; STORE SECOND PORTION HIGH
7733 00002A3D 6683C750          <1>       ADD    DI, 80           ; POINT TO NEXT LOCATION
7734 00002A41 FECE              <1>       DEC    DH
7735 00002A43 75C4              <1>       JNZ    short S8         ; KEEP GOING
7736 00002A45 5E                <1>       POP    eSI              ; RECOVER CODE POINTER
7737 00002A46 5F                <1>       POP    eDI              ; RECOVER REGEN POINTER
7738 00002A47 47                <1>       INC    eDI              ; POINT TO NEXT CHAR POSITION
7739 00002A48 47                <1>       INC    eDI
7740 00002A49 E2BA              <1>       LOOP   S7               ; MORE TO WRITE
7741 00002A4B C3                <1>       retn
7742                            <1>
7743                            <1> ; 04/07/2016
7744                            <1> ; 01/07/2016
7745                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7746                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7747                            <1> ;----------------------------------------
7748                            <1> ; GRAPHICS READ
7749                            <1> ;----------------------------------------
7750                            <1> GRAPHICS_READ:
7751 00002A4C E8A1000000        <1>       CALL   S26              ; CONVERTED TO OFFSET IN REGEN
7752 00002A51 89C6              <1>       MOV    eSI, eAX         ; SAVE IN SI
7753 00002A53 81C600800B00      <1>       add    esi, 0B8000h     ; 01/07/2016
7754 00002A59 83EC08            <1>       SUB    eSP, 8           ; ALLOCATE SPACE FOR THE READ CODE POINT
7755 00002A5C 89E5              <1>       MOV    eBP, eSP         ; POINTER TO SAVE AREA
7756                            <1>
7757                            <1> ;-----      DETERMINE GRAPHICS MODES
7758 00002A5E B604              <1>       mov    dh, 4            ; number of passes ; 01/07/2016
7759 00002A60 803D[9A6E0000]06  <1>       CMP    byte [CRT_MODE], 6
7760 00002A67 7219              <1>       JC     short S12        ; MEDIUM RESOLUTION
7761                            <1>
7762                            <1> ;-----      HIGH RESOLUTION READ
7763                            <1> ;-----      GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
7764                            <1>       ;MOV   DH,4             ; NUMBER OF PASSES
7765                            <1> S11:
7766 00002A69 8A06              <1>       MOV    AL, [eSI]        ; GET FIRST BYTE
7767 00002A6B 884500            <1>       MOV    [eBP], AL        ; SAVE IN STORAGE AREA
7768 00002A6E 45                <1>       INC    eBP              ; NEXT LOCATION
7769 00002A6F 8A8600200000      <1>       MOV    AL, [eSI+2000H]         ; GET LOWER REGION BYTE
7770 00002A75 884500            <1>       MOV    [eBP], AL        ; ADJUST AND STORE
7771 00002A78 45                <1>       INC    eBP
7772 00002A79 83C650            <1>       ADD    eSI, 80                 ; POINTER INTO REGEN
7773 00002A7C FECE              <1>       DEC    DH               ; LOOP CONTROL
7774 00002A7E 75E9              <1>       JNZ    short S11        ; DO IT SOME MORE
7775 00002A80 EB1D              <1>       JMP    SHORT S14        ; GO MATCH THE SAVED CODE POINTS
7776                            <1>
7777                            <1> ;-----      MEDIUM RESOLUTION READ
7778                            <1> S12:
7779 00002A82 66D1E6            <1>       SAL    SI, 1            ; OFFSET*2 SINCE 2 BYTES/CHAR
7780                            <1>       ;MOV   DH, 4            ; NUMBER OF PASSES
7781                            <1> S13:
7782 00002A85 E84B000000        <1>       CALL   S23              ; GET BYTES FROM REGEN INTO SINGLE SAVE
7783 00002A8A 81C6FE1F0000      <1>       ADD    eSI, 2000H-2     ; GO TO LOWER REGION
7784 00002A90 E840000000        <1>       CALL   S23              ; GET THIS PAIR INTO SAVE
7785 00002A95 81EEB21F0000      <1>       SUB    eSI, 2000H-80+2         ; ADJUST POINTER BACK INTO UPPER
7786 00002A9B FECE              <1>       DEC    DH
7787 00002A9D 75E6              <1>       JNZ    short S13        ; KEEP GOING UNTIL ALL 8 DONE
7788                            <1>
7789                            <1> ;-----      SAVE AREA HAS CHARACTER IN IT, MATCH IT
7790                            <1> S14:                          ; FIND_CHAR
7791 00002A9F BF[845F0100]      <1>       MOV    eDI, CRT_CHAR_GEN ; ESTABLISH ADDRESSING
7792 00002AA4 83ED08            <1>       SUB    eBP, 8           ; ADJUST POINTER TO START OF SAVE AREA
7793 00002AA7 89EE              <1>       MOV    eSI, eBP
7794                            <1> S15:
7795 00002AA9 66B80001          <1>       mov    ax, 256          ; NUMBER TO TEST AGAINST
7796                            <1> S16:
7797 00002AAD 56                <1>       PUSH   eSI              ; SAVE SAVE AREA POINTER
7798 00002AAE 57                <1>       PUSH   eDI              ; SAVE CODE POINTER
7799                            <1>       ;MOV   eCX, 4           ; NUMBER OF WORDS TO MATCH
7800                            <1>       ;REPE  CMPSW            ; COMPARE THE 8 BYTES AS WORDS
7801 00002AAF A7                <1>       cmpsd                   ; compare first 4 bytes
7802 00002AB0 7501              <1>       jne    short S17        ;
7803 00002AB2 A7                <1>       cmpsd                   ; compare last 4 bytes
7804                            <1> S17:
7805 00002AB3 5F                <1>       POP    eDI              ; RECOVER THE POINTERS
7806 00002AB4 5E                <1>       POP    eSI
7807                            <1>       ;JZ    short S18        ; IF ZERO FLAG SET, THEN MATCH OCCURRED
7808 00002AB5 7407              <1>       je     short S18
7809                            <1>       ;                       ; NO MATCH, MOVE ON TO NEXT
7810 00002AB7 83C708            <1>       ADD    eDI, 8           ; NEXT CODE POINT
7811 00002ABA 6648              <1>       dec    ax               ; LOOP CONTROL
7812 00002ABC 75EF              <1>       JNZ    short S16        ; DO ALL OF THEM
7813                            <1>
7814                            <1> ;-----      CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
7815                            <1> S18:
7816 00002ABE 83C408            <1>       ADD    eSP, 8           ; READJUST THE STACK, THROW AWAY SAVE
7817 00002AC1 C3                <1>       retn                    ; ALL DONE
7818                            <1>
7819                            <1> ; 12/04/2021
7820                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7821                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7822                            <1> ;----------------------------------------------
7823                            <1> ; EXPAND BYTE
7824                            <1> ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
7825                            <1> ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
7826                            <1> ;  THE RESULT IS LEFT IN AX
7827                            <1> ;----------------------------------------------
7828                            <1> S21:
```

```
7829                                    <1>       ;PUSH  CX                 ; SAVE REGISTER
7830                                    <1>       ; 12/04/2021
7831 00002AC2 51                        <1>       push   ecx
7832                                    <1>       ;MOV   CX, 8              ; SHIFT COUNT REGISTER FOR ONE BYTE
7833 00002AC3 B108                      <1>       mov    cl, 8
7834                                    <1> S22:
7835 00002AC5 D0C8                      <1>       ROR    AL,1               ; SHIFT BITS, LOW BIT INTO CARRY FLAG
7836 00002AC7 66D1DD                    <1>       RCR    BP,1               ; MOVE CARRY FLAG (LOW BIT INTO RESULTS
7837 00002ACA 66D1FD                    <1>       SAR    BP,1               ; SIGN EXTEND HIGH BIT (DOUBLE IT)
7838                                    <1>       ;LOOP  S22                ; REPEAT FOR ALL 8 BITS
7839 00002ACD FEC9                      <1>       dec    cl
7840 00002ACF 75F4                      <1>       jnz    short S22
7841 00002AD1 6695                      <1>       XCHG   AX, BP             ; MOVE RESULTS TO PARAMETER REGISTER
7842                                    <1>       ;POP   CX                 ; RECOVER REGISTER
7843                                    <1>       ; 12/04/2021
7844 00002AD3 59                        <1>       pop    ecx
7845 00002AD4 C3                        <1>       RETn                      ; ALL DONE
7846                                    <1>
7847                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7848                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7849                                    <1> ;-------------------------------------------------
7850                                    <1> ; MED_READ_BYTE
7851                                    <1> ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
7852                                    <1> ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
7853                                    <1> ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
7854                                    <1> ;  POSITION IN THE SAVE AREA
7855                                    <1> ; ENTRY --
7856                                    <1> ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
7857                                    <1> ;  BX = EXPANDED FOREGROUND COLOR
7858                                    <1> ;  BP = POINTER TO SAVE AREA
7859                                    <1> ; EXIT --
7860                                    <1> ;  SI AND BP ARE INCREMENTED
7861                                    <1> ;-------------------------------------------------
7862                                    <1> S23:
7863 00002AD5 66AD                      <1>       LODSW                     ; GET FIRST BYTE AND SECOND BYTES
7864 00002AD7 86C4                      <1>       XCHG   AL, AH             ; SWAP FOR COMPARE
7865 00002AD9 66B900C0                  <1>       MOV    CX, 0C000H         ; 2 BIT MASK TO TEST THE ENTRIES
7866 00002ADD B200                      <1>       MOV    DL, 0              ; RESULT REGISTER
7867                                    <1> S24:
7868 00002ADF 6685C8                    <1>       TEST   AX, CX             ; IS THIS SECTION BACKCROUND?
7869 00002AE2 7401                      <1>       JZ    short S25           ; IF ZERO, IT IS BACKGROUND (CARRY=0)
7870 00002AE4 F9                        <1>       STC                       ; WASN'T, SO SET CARRY
7871                                    <1> S25:
7872 00002AE5 D0D2                      <1>       RCL    DL, 1              ; MOVE THAT BIT INTO THE RESULT
7873 00002AE7 66C1E902                  <1>       SHR    CX, 2              ; MOVE THE MASK TO THE RIGHT BY 2 BITS
7874 00002AEB 73F2                      <1>       JNC    short S24          ; DO IT AGAIN IF MASK DIDN'T FALL OUT
7875 00002AED 885500                    <1>       MOV    [eBP], DL          ; STORE RESULT IN SAVE AREA
7876 00002AF0 45                        <1>       INC    eBP                ; ADJUST POINTER
7877 00002AF1 C3                        <1>       RETn                      ; ALL DONE
7878                                    <1>
7879                                    <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7880                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7881                                    <1> ;-----------------------------------------
7882                                    <1> ; V4_POSITION
7883                                    <1> ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
7884                                    <1> ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
7885                                    <1> ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
7886                                    <1> ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
7887                                    <1> ;  BE DOUBLED.
7888                                    <1> ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
7889                                    <1> ; EXIT--
7890                                    <1> ;  AX CONTAINS OFFSET INTO REGEN BUFFER
7891                                    <1> ;-----------------------------------------
7892                                    <1> S26:
7893 00002AF2 0FB705[0E890100]          <1>       movzx  eax, word [CURSOR_POSN] ; GET CURRENT CURSOR
7894                                    <1> GRAPH_POSN:
7895 00002AF9 53                        <1>       PUSH   eBX                ; SAVE REGISTER
7896 00002AFA 0FB6D8                    <1>       movzx  ebx, al            ; SAVE A COPY OF CURRENT CURSOR
7897 00002AFD A0[9C6E0000]              <1>       MOV    AL, [CRT_COLS]     ; GET BYTES PER COLUMN
7898 00002B02 F6E4                      <1>       MUL    AH                 ; MULTIPLY BY ROWS
7899 00002B04 66C1E002                  <1>       SHL    AX, 2              ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
7900 00002B08 01D8                      <1>       ADD    eAX, eBX           ; DETERMINE OFFSET
7901 00002B0A 5B                        <1>       POP    eBX                ; RECOVER POINTER
7902 00002B0B C3                        <1>       RETn                      ; ALL DONE
7903                                    <1>
7904                                    <1> ; 09/07/2016
7905                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7906                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7907                                    <1> ;-----------------------------------------------
7908                                    <1> ; SET_COLOR
7909                                    <1> ;     THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
7910                                    <1> ;     AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
7911                                    <1> ; INPUT
7912                                    <1> ;     (BH) HAS COLOR ID
7913                                    <1> ;           IF BH=0, THE BACKGROUND COLOR VALUE IS SET
7914                                    <1> ;                 FROM THE LOW BITS OF BL (0-31)
7915                                    <1> ;           IF BH=1, THE PALETTE SELECTION IS MADE
7916                                    <1> ;                 BASED ON THE LOW BIT OF BL:
7917                                    <1> ;                       0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
7918                                    <1> ;                       1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
7919                                    <1> ;     (BL) HAS THE COLOR VALUE TO BE USED
7920                                    <1> ; OUTPUT
7921                                    <1> ;     THE COLOR SELECTION IS UPDATED
7922                                    <1> ;-----------------------------------------------
7923                                    <1> SET_COLOR:
7924 00002B0C 803D[9A6E0000]07          <1>       cmp   byte [CRT_MODE], 7  ; 09/07/2016
7925 00002B13 0F87D8EFFFFF              <1>       ja     VIDEO_RETURN       ; nothing to do for VGA modes
7926                                    <1>
7927                                    <1>       ;MOV   DX, [ADDR_6845]    ; I/O PORT FOR PALETTE
7928                                    <1>       ;mov   dx, 3D4h
7929                                    <1>       ;ADD   DX,5               ; OVERSCAN PORT
7930 00002B19 66BAD903                  <1>       mov    dx, 3D9h
7931 00002B1D A0[9D6E0000]              <1>       MOV    AL, [CRT_PALETTE]  ; GET THE CURRENT PALETTE VALUE
7932 00002B22 08FF                      <1>       OR     BH, BH             ; IS THIS COLOR 0?
7933 00002B24 7512                      <1>       JNZ    short M20          ; OUTPUT COLOR 1
```

```
                                <1>
                                <1> ;-----      HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
                                <1>
00002B26 24E0                   <1>       AND   AL, 0E0H            ; TURN OFF LOW 5 BITS OF CURRENT
00002B28 80E31F                 <1>       AND   BL, 01FH            ; TURN OFF HIGH 3 BITS OF INPUT VALUE
00002B2B 08D8                   <1>       OR    AL, BL              ; PUT VALUE INTO REGISTER
                                <1> M19:                            ; OUTPUT THE PALETTE
00002B2D EE                     <1>       OUT   DX, AL              ; OUTPUT COLOR SELECTION TO 3D9 PORT
00002B2E A2[9D6E0000]           <1>       MOV   [CRT_PALETTE], AL   ; SAVE THE COLOR VALUE
00002B33 E9B9EFFFFF             <1>       JMP   VIDEO_RETURN
                                <1>
                                <1> ;-----      HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
                                <1>
                                <1> M20:
00002B38 24DF                   <1>       AND   AL, 0DFH            ; TURN OFF PALETTE SELECT BIT
00002B3A D0EB                   <1>       SHR   BL, 1               ; TEST THE LOW ORDER BIT OF BL
00002B3C 73EF                   <1>       JNC   short M19           ; ALREADY DONE
00002B3E 0C20                   <1>       OR    AL, 20H                   ; TURN ON PALETTE SELECT BIT
00002B40 EBEB                   <1>       JMP   short M19           ; GO DO IT
                                <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> ; READ DOT -- WRITE DOT
                                <1> ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
                                <1> ;  DOT AT THE INDICATED LOCATION
                                <1> ; ENTRY --
                                <1> ;   DX = ROW (0-199)      (THE ACTUAL VALUE DEPENDS ON THE MODE)
                                <1> ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
                                <1> ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
                                <1> ;     REQUIRED FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
                                <1> ;     BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
                                <1> ;   DS = DATA SEGMENT
                                <1> ;   ES = REGEN SEGMENT
                                <1> ;
                                <1> ; EXIT
                                <1> ;     AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
                                <1> ;--------------------------------------------
                                <1>
                                <1> READ_DOT:
                                <1>       ; 09/07/2016
00002B42 8A25[9A6E0000]         <1>       mov   ah,  [CRT_MODE]
00002B48 80FC07                 <1>       cmp   ah, 7 ; 6!?
00002B4B 760A                   <1>       jna   short read_dot_cga
                                <1>
00002B4D E8D1030000             <1>       call  vga_read_pixel
                                <1>       ; al = pixel value
00002B52 E99FEFFFFF             <1>       jmp   _video_return
                                <1>
                                <1> read_dot_cga:
                                <1>       ;je   VIDEO_RETURN ; 7
00002B57 80FC04                 <1>       cmp   ah, 4 ; graphics ?
00002B5A 0F8291EFFFFF           <1>       jb    VIDEO_RETURN ; no, text mode, nothing to do
                                <1>
00002B60 E853000000             <1>       CALL  R3                  ; DETERMINE BYTE POSITION OF DOT
00002B65 8A06                   <1>       MOV   AL, [eSI]           ; GET THE BYTE
00002B67 20E0                   <1>       AND   AL, AH              ; MASK OFF THE OTHER BITS IN THE BYTE
00002B69 D2E0                   <1>       SHL   AL, CL              ; LEFT JUSTIFY THE VALUE
00002B6B 88F1                   <1>       MOV   CL, DH              ; GET NUMBER OF BITS IN RESULT
00002B6D D2C0                   <1>       ROL   AL, CL              ; RIGHT JUSTIFY THE RESULT
                                <1>       ;JMP  VIDEO_RETURN        ; RETURN FROM VIDEO I/O
00002B6F 0FB6C0                 <1>       movzx eax, al
00002B72 E97FEFFFFF             <1>       jmp   _video_return
                                <1>
                                <1> ; 12/04/2021
                                <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> WRITE_DOT:
                                <1>       ; 09/07/2016
00002B77 8A25[9A6E0000]         <1>       mov   ah, [CRT_MODE]
00002B7D 80FC07                 <1>       cmp   ah, 7 ; 6!?
00002B80 760A                   <1>       jna   short write_dot_cga
                                <1>
00002B82 E80B030000             <1>       call  vga_write_pixel
00002B87 E965EFFFFF             <1>       jmp   VIDEO_RETURN
                                <1>
                                <1> write_dot_cga:
                                <1>       ;je   VIDEO_RETURN ; 7
00002B8C 80FC04                 <1>       cmp   ah, 4 ; graphics ?
00002B8F 0F825CEFFFFF           <1>       jb    VIDEO_RETURN ; no, text mode, nothing to do
                                <1>
                                <1>       ;;PUSH AX                 ; SAVE DOT VALUE
                                <1>       ;PUSH  AX                 ; TWICE
                                <1>       ; 12/04/2021
00002B95 50                     <1>       push  eax
00002B96 E81D000000             <1>       CALL  R3                  ; DETERMINE BYTE POSITION OF THE DOT
00002B9B D2E8                   <1>       SHR   AL, CL              ; SHIFT TO SET UP THE BITS FOR OUTPUT
00002B9D 20E0                   <1>       AND   AL, AH              ; STRIP OFF THE OTHER BITS
00002B9F 8A0E                   <1>       MOV   CL, [eSI]           ; GET THE CURRENT BYTE
                                <1>       ;POP  BX                  ; RECOVER XOR FLAG
                                <1>       ; 12/04/2021
00002BA1 5B                     <1>       pop   ebx
00002BA2 F6C380                 <1>       TEST  BL, 80H                  ; IS IT ON
00002BA5 750D                   <1>       JNZ   short R2            ; YES, XOR THE DOT
00002BA7 F6D4                   <1>       NOT   AH                  ; SET MASK TO REMOVE THE INDICATED BITS
00002BA9 20E1                   <1>       AND   CL, AH
00002BAB 08C8                   <1>       OR    AL, CL              ; OR IN THE NEW VALUE OF THOSE BITS
                                <1> R1:                             ; FINISH_DOT
00002BAD 8806                   <1>       MOV   [eSI], AL           ; RESTORE THE BYTE IN MEMORY
                                <1>       ;;POP  AX
00002BAF E93DEFFFFF             <1>       JMP   VIDEO_RETURN        ; RETURN FROM VIDEO I/O
                                <1> R2:                             ; XOR_DOT
```

```
8039 00002BB4 30C8                <1>       XOR    AL, CL              ; EXCLUSIVE OR THE DOTS
8040 00002BB6 EBF5                <1>       JMP    short R1            ; FINISH UP THE WRITING
8041                              <1>
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> ;--------------------------------------------
8046                              <1> ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
8047                              <1> ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
8048                              <1> ; ENTRY --
8049                              <1> ;  DX = ROW VALUE (0-199)
8050                              <1> ;  CX = COLUMN VALUE (0-639)
8051                              <1> ; EXIT --
8052                              <1> ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
8053                              <1> ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
8054                              <1> ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
8055                              <1> ;  DH = # BITS IN RESULT
8056                              <1> ;  BX = MODIFIED
8057                              <1> ;--------------------------------------------
8058                              <1> R3:
8059                              <1>
8060                              <1> ;-----      DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
8061                              <1> ;-----       ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
8062                              <1>
8063 00002BB8 0FB7F0              <1>       movzx  esi, ax             ; WILL SAVE AL AND AH DURING OPERATION
8064 00002BBB B028                <1>       MOV    AL, 40
8065 00002BBD F6E2                <1>       MUL    DL                  ; AX= ADDRESS OF START OF INDICATED ROW
8066 00002BBF A808                <1>       TEST   AL, 08H             ; TEST FOR EVEN/ODD ROW CALCULATED
8067 00002BC1 7404                <1>       JZ     short R4            ; JUMP IF EVEN ROW
8068 00002BC3 6605D81F            <1>       ADD    AX, 2000H-40        ; OFFSET TO LOCATION OF ODD ROWS ADJUST
8069                              <1> R4:                               ; EVEN_ROW
8070 00002BC7 6696                <1>       XCHG   SI, AX              ; MOVE POINTER TO (SI) AND RECOVER (AX)
8071 00002BC9 81C600800B00        <1>       add    esi, 0B8000h
8072 00002BCF 6689CA              <1>       MOV    DX, CX              ; COLUMN VALUE TO DX
8073                              <1>
8074                              <1> ;-----      DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
8075                              <1>
8076                              <1> ; SET UP THE REGISTERS ACCORDING TO THE MODE
8077                              <1> ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
8078                              <1> ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
8079                              <1> ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
8080                              <1> ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
8081                              <1>
8082 00002BD2 66BBC002            <1>       MOV    BX, 2C0H
8083 00002BD6 66B90203            <1>       MOV    CX, 302H            ; SET PARMS FOR MED RES
8084 00002BDA 803D[9A6E0000]06    <1>       CMP    byte [CRT_MODE], 6
8085 00002BE1 7208                <1>       JC     short R5            ; HANDLE IF MED RES
8086 00002BE3 66BB8001            <1>       MOV    BX, 180H
8087 00002BE7 66B90307            <1>       MOV    CX, 703H            ; SET PARMS FOR HIGH RES
8088                              <1>
8089                              <1> ;-----      DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
8090                              <1> R5:
8091 00002BEB 20D5                <1>       AND    CH, DL              ; ADDRESS OF PEL WITHIN BYTE TO CH
8092                              <1>
8093                              <1> ;-----      DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
8094                              <1>
8095 00002BED 66D3EA              <1>       SHR    DX, CL              ; SHIFT BY CORRECT AMOUNT
8096 00002BF0 6601D6              <1>       ADD    SI, DX              ; INCREMENT THE POINTER
8097 00002BF3 88FE                <1>       MOV    DH, BH              ; GET THE # OF BITS IN RESULT TO DH
8098                              <1>
8099                              <1> ;-----      MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
8100                              <1>
8101 00002BF5 28C9                <1>       SUB    CL, CL              ; ZERO INTO STORAGE LOCATION
8102                              <1> R6:
8103 00002BF7 D0C8                <1>       ROR    AL, 1               ; LEFT JUSTIFY VALUE IN AL (FOR WRITE)
8104 00002BF9 00E9                <1>       ADD    CL, CH              ; ADD IN THE BIT OFFSET VALUE
8105 00002BFB FECF                <1>       DEC    BH                  ; LOOP CONTROL
8106 00002BFD 75F8                <1>       JNZ    short R6            ; ON EXIT, CL HAS COUNT TO RESTORE BITS
8107 00002BFF 88DC                <1>       MOV    AH, BL              ;  GET MASK TO AH
8108 00002C01 D2EC                <1>       SHR    AH, CL              ;  MOVE THE MASK TO CORRECT LOCATION
8109 00002C03 C3                  <1>       RETn                       ;  RETURN WITH EVERYTHING SET UP
8110                              <1>
8111                              <1> load_dac_palette:
8112                              <1>       ; 29/07/2016
8113                              <1>       ; 23/07/2016
8114                              <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
8115                              <1>       ; (set_mode_vga)
8116                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8117                              <1>       ; vgabios-0.7a (2011)
8118                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8119                              <1>       ; 'vgabios.c', 'load_dac_palette'
8120                              <1>       ;
8121                              <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
8122                              <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
8123                              <1>       ;
8124                              <1>       ; INPUT -> AH = DAC selection number (3, 2 or 1)
8125                              <1>       ; OUTPUT -> ECX = 0, AX = 0
8126                              <1>       ; (Modifed registers: EAX, ECX, EDX, ESI)
8127                              <1>       ;
8128 00002C04 66BAC803            <1>       mov    dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
8129 00002C08 28C0                <1>       sub    al, al ; 0
8130 00002C0A EE                  <1>       out    dx, al ; 0 ; color index, always 0 at the beginning
8131 00002C0B 6642                <1>       inc    dx   ; 3C9h ; VGAREG_DAC_DATA
8132 00002C0D B900010000          <1>       mov    ecx, 256   ; always 256*3 values
8133                              <1>       ;push  esi
8134 00002C12 88E0                <1>       mov    al, ah
8135 00002C14 B43F                <1>       mov    ah, 3Fh      ; 3Fh except DAC selection number 3
8136 00002C16 3C02                <1>       cmp    al, 2
8137 00002C18 7414                <1>       je     short l_dac_p_2
8138 00002C1A 7719                <1>       ja     short l_dac_p_3
8139 00002C1C 20C0                <1>       and    al, al
8140 00002C1E 7507                <1>       jnz    short l_dac_p_1
8141                              <1> l_dac_p_0:
8142 00002C20 BE[445A0100]        <1>       mov    esi, palette0
8143 00002C25 EB15                <1>       jmp    short l_dac_p_4
```

```
                                <1> l_dac_p_1:
00002C27 BE[045B0100]           <1>         mov     esi, palette1
00002C2C EB0E                   <1>         jmp     short l_dac_p_4
                                <1> l_dac_p_2:
00002C2E BE[C45B0100]           <1>         mov     esi, palette2
00002C33 EB07                   <1>         jmp     short l_dac_p_4
                                <1> l_dac_p_3:
00002C35 B4FF                   <1>         mov     ah, 0FFh ; dac registers
00002C37 BE[845C0100]           <1>         mov     esi, palette3
                                <1> l_dac_p_4:
00002C3C AC                     <1>         lodsb
00002C3D EE                     <1>         out     dx, al  ; Red
00002C3E AC                     <1>         lodsb
00002C3F EE                     <1>         out     dx, al ; Green
00002C40 AC                     <1>         lodsb
00002C41 EE                     <1>         out     dx, al ; Blue
00002C42 20E4                   <1>         and     ah, ah
00002C44 7405                   <1>         jz      short l_dac_p_5
00002C46 FECC                   <1>         dec     ah
00002C48 E2F2                   <1>         loop    l_dac_p_4
                                <1>         ;pop    esi
00002C4A C3                     <1>         retn
                                <1> l_dac_p_5:
                                <1>         ; 29/07/2016
00002C4B FEC9                   <1>         dec     cl
00002C4D 7407                   <1>         jz      short l_dac_p_7
                                <1>         ;
00002C4F 28C0                   <1>         sub     al, al ; 0
                                <1> l_dac_p_6:
00002C51 EE                     <1>         out     dx, al ; outb(VGAREG_DAC_DATA,0);
00002C52 EE                     <1>         out     dx, al
00002C53 EE                     <1>         out     dx, al
00002C54 E2FB                   <1>         loop    l_dac_p_6
                                <1> l_dac_p_7:
                                <1>         ;pop    esi
00002C56 C3                     <1>         retn
                                <1>
                                <1> gray_scale_summing:
                                <1>         ; 12/04/2021
                                <1>         ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; (set_mode_vga)
                                <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_perform_gray_scale_summing'
                                <1>         ;
                                <1>         ; Oracle VirtualBox 5.0.24 VGABios Source Code
                                <1>         ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                                <1>         ;
                                <1>
                                <1>         ; INPUT -> EBX = Start address (color index <= 255)
                                <1>         ;         ECX = Count (<= 256)
                                <1>         ; OUTPUT -> (E)CX = 0
                                <1>         ; (Modifed registers: EAX, ECX, EDX, EBX)
                                <1>
00002C57 66BADA03               <1>         mov     dx, 3DAh ; VGAREG_ACTL_RESET
00002C5B EC                     <1>         in      al, dx
00002C5C 30C0                   <1>         xor     al, al ; 0
00002C5E 66BAC003               <1>         mov     dx, 3C0h ; VGAREG_ACTL_ADDRESS
00002C62 EE                     <1>         out     dx, al ; clear bit 5
                                <1>                        ; (while loading palette registers)
                                <1>         ; set read address and switch to read mode
                                <1> g_s_s_1:
00002C63 66BAC703               <1>         mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
00002C67 88D8                   <1>         mov     al, bl
00002C69 EE                     <1>         out     dx, al
                                <1>         ; get 6-bit wide RGB data values
                                <1>         ; intensity = (0.3*Red)+(0.59*Green)+(0.11*Blue)
                                <1>         ; i = ( ( 77*r + 151*g + 28*b ) + 0x80 ) >> 8;
00002C6A 66BAC903               <1>         mov     dx, 3C9h ; VGAREG_DAC_DATA
00002C6E EC                     <1>         in      al, dx ; red
00002C6F B44D                   <1>         mov     ah, 77 ; 0.3* Red
00002C71 F6E4                   <1>         mul     ah
                                <1>         ;push  ax
                                <1>         ; 12/04/2021
00002C73 50                     <1>         push    eax
00002C74 EC                     <1>         in      al, dx ; green
00002C75 B497                   <1>         mov     ah, 151  ; 0.59 * Green
00002C77 F6E4                   <1>         mul     ah
                                <1>         ;push  ax
                                <1>         ; 12/04/2021
00002C79 50                     <1>         push    eax
00002C7A EC                     <1>         in      al, dx ; blue
00002C7B B41C                   <1>         mov     ah, 28 ; 0.11 * Blue
00002C7D F6E4                   <1>         mul     ah
                                <1>         ;pop   dx
                                <1>         ; 12/04/2021
00002C7F 5A                     <1>         pop     edx
00002C80 6601D0                 <1>         add     ax, dx
                                <1>         ;pop   dx
                                <1>         ; 12/04/2021
00002C83 5A                     <1>         pop     edx
00002C84 6601D0                 <1>         add     ax, dx
00002C87 66058000               <1>         add     ax, 80h
00002C8B B03F                   <1>         mov     al, 3Fh
00002C8D 38C4                   <1>         cmp     ah, al ; if(i>0x3f)i=0x3f
00002C8F 7602                   <1>         jna     short g_s_s_2
00002C91 88C4                   <1>         mov     ah, al
                                <1> g_s_s_2:
00002C93 66BAC803               <1>         mov     dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
00002C97 88D8                   <1>         mov     al, bl ; color index
00002C99 EE                     <1>         out     dx, al
00002C9A 88E0                   <1>         mov     al, ah ; intensity
00002C9C 6642                   <1>         inc     dx ; 3C9h ; VGAREG_DAC_DATA
00002C9E EE                     <1>         out     dx, al ; R (R=G=B)
```

```
8249 00002C9F 88E0              <1>       mov   al, ah ; intensity
8250 00002CA1 EE                <1>       out   dx, al ; G (R=G=B)
8251 00002CA2 88E0              <1>       mov   al, ah ; intensity
8252 00002CA4 EE                <1>       out   dx, al ; B (R=G=B)
8253 00002CA5 6649              <1>       dec   cx
8254 00002CA7 7404              <1>       jz    short g_s_s_3
8255 00002CA9 FEC3              <1>       inc   bl    ; next color index value
8256 00002CAB EBB6              <1>       jmp   short g_s_s_1
8257                            <1> g_s_s_3:
8258 00002CAD 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
8259 00002CB1 EC                <1>       in    al, dx
8260 00002CB2 B020              <1>       mov   al, 20h
8261 00002CB4 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
8262 00002CB8 EE                <1>       out   dx, al ; 20h -> set bit 5
8263                            <1>                    ; (after loading palette regs)
8264 00002CB9 C3                <1>       retn
8265                            <1>
8266                            <1> vga_write_char_attr:
8267                            <1> vga_write_char_only:
8268                            <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
8269                            <1>       ;
8270                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8271                            <1>       ; vgabios-0.7a (2011)
8272                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8273                            <1>       ; 'vgabios.c', 'biosfn_write_char_attr'
8274                            <1>       ; 'biosfn_write_char_only'
8275                            <1>
8276                            <1>       ; INPUT ->
8277                            <1>       ; [CRT_MODE] = current video mode (>7)
8278                            <1>       ; CX = Count of characters to write
8279                            <1>       ; AL = Character to write
8280                            <1>       ; BL = Color of character
8281                            <1>       ; OUTPUT ->
8282                            <1>       ; Regen buffer updated
8283                            <1>
8284 00002CBA 8A25[9A6E0000]    <1>       mov   ah, [CRT_MODE]
8285 00002CC0 668B15[0E890100]  <1>       mov   dx, [CURSOR_POSN] ; cursor pos for page 0
8286                            <1>
8287 00002CC7 BE[B66E0000]      <1>       mov   esi, vga_modes
8288 00002CCC 89F7              <1>       mov   edi, esi
8289 00002CCE 83C710            <1>       add   edi, vga_mode_count
8290                            <1> vga_wca_0:
8291 00002CD1 AC                <1>       lodsb
8292 00002CD2 38E0              <1>       cmp   al, ah ; [CRT_MODE]
8293 00002CD4 7405              <1>       je    short vga_wca_2
8294 00002CD6 39FE              <1>       cmp   esi, edi
8295 00002CD8 72F7              <1>       jb    short vga_wca_0
8296                            <1> vga_wca_1:
8297 00002CDA C3                <1>       retn  ; nothing to do
8298                            <1> vga_wca_2:
8299 00002CDB 83C64F            <1>       add   esi, vga_memmodel - (vga_modes + 1)
8300                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8301                            <1>
8302                            <1>       ; biosfn_write_char_attr (car,page,attr,count)
8303                            <1>       ; AL = car, page = 0, BL = attr, CX = count
8304 00002CDE 803E04            <1>       cmp   byte [esi], PLANAR4
8305 00002CE1 741D              <1>       je    short vga_wca_planar
8306 00002CE3 803E03            <1>       cmp   byte [esi], PLANAR1
8307 00002CE6 7418              <1>       je    short vga_wca_planar
8308                            <1> vga_wca_linear8:
8309                            <1>       ; while((count-->0) && (xcurs<nbcols))
8310                            <1>       ; CX = count
8311 00002CE8 6621C9            <1>       and   cx, cx
8312 00002CEB 74ED              <1>       jz    short vga_wca_1
8313 00002CED 3A15[9C6E0000]    <1>       cmp   dl, [CRT_COLS]
8314 00002CF3 73E5              <1>       jnb   short vga_wca_1
8315                            <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
8316                            <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
8317                            <1>       ; [CRT_COLS] = nbcols
8318 00002CF5 E81E000000        <1>       call  write_gfx_char_lin
8319 00002CFA 6649              <1>       dec   cx ; count
8320 00002CFC FEC2              <1>       inc   dl ; xcurs
8321 00002CFE EBE8              <1>       jmp   short vga_wca_linear8
8322                            <1> vga_wca_planar:
8323                            <1>       ; while((count-->0) && (xcurs<nbcols))
8324                            <1>       ; CX = count
8325 00002D00 6621C9            <1>       and   cx, cx
8326 00002D03 74D5              <1>       jz    short vga_wca_1
8327 00002D05 3A15[9C6E0000]    <1>       cmp   dl, [CRT_COLS]
8328 00002D0B 73CD              <1>       jnb   short vga_wca_1
8329                            <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
8330                            <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
8331                            <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
8332 00002D0D E8A9000000        <1>       call  write_gfx_char_pl4
8333 00002D12 6649              <1>       dec   cx ; count
8334 00002D14 FEC2              <1>       inc   dl ; xcurs
8335 00002D16 EBE8              <1>       jmp   short vga_wca_planar
8336                            <1>
8337                            <1> write_gfx_char_lin:
8338                            <1>       ; 08/01/2021
8339                            <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
8340                            <1>       ; 08/08/2016
8341                            <1>       ; 31/07/2016
8342                            <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
8343                            <1>       ;
8344                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8345                            <1>       ; vgabios-0.7a (2011)
8346                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8347                            <1>       ; 'vgabios.c', 'write_gfx_char_lin'
8348                            <1>
8349                            <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols)
8350                            <1>       ; INPUT ->
8351                            <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
8352                            <1>       ; [CRT_COLS] = nbcols
8353                            <1>       ; OUTPUT ->
```

```
8354                                    <1>       ; Regen buffer updated
8355                                    <1>
8356 00002D18 51                        <1>       push   ecx
8357 00002D19 53                        <1>       push   ebx
8358 00002D1A 52                        <1>       push   edx
8359 00002D1B 50                        <1>       push   eax
8360                                    <1>       ; addr=xcurs*8+ycurs*nbcols*64;
8361                                    <1>       ; 08/08/2016
8362 00002D1C 0FB6F0                    <1>       movzx  esi, al ; car
8363                                    <1>       ; 08/01/2021
8364                                    <1>       ;movzx eax, dh ; ycurs
8365                                    <1>       ;mov   ah, [CRT_COLS] ; nbcols
8366                                    <1>       ;mul   ah
8367 00002D1F A0[9C6E0000]             <1>       mov    al, [CRT_COLS]
8368 00002D24 F6E6                      <1>       mul    dh
8369                                    <1>       ;shl   ax, 6 ; * 64
8370 00002D26 66C1E003                  <1>       shl    ax, 3 ; 8 * ycurs * [CRT_COLS]
8371                                    <1>       ;sub   dh, dh
8372                                    <1>       ;shl   dx, 3 ; xcurs * 8
8373                                    <1>       ;movzx edi, dx
8374 00002D2A BF00000A00                <1>       mov    edi, 0A0000h
8375 00002D2F 30F6                      <1>       xor    dh, dh
8376 00002D31 6689D7                    <1>       mov    di, dx
8377                                    <1>       ;movzx edi, dl
8378 00002D34 66C1E703                  <1>       shl    di, 3 ; xcurs * 8
8379                                    <1>       ;xor   dh, dh
8380 00002D38 8A15[9E6E0000]            <1>       mov    dl, [CHAR_HEIGHT]
8381 00002D3E 66F7E2                    <1>       mul    dx
8382                                    <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
8383                                    <1>       ;add   edi, eax ; addr
8384                                    <1>       ;add   edi, 0A0000h
8385 00002D41 6601C7                    <1>       add    di, ax
8386                                    <1>       ;shl   si, 3 ; car * 8
8387 00002D44 30E4                      <1>       xor    ah, ah
8388 00002D46 A0[9E6E0000]              <1>       mov    al, [CHAR_HEIGHT]
8389 00002D4B 66F7E6                    <1>       mul    si
8390 00002D4E 6689C6                    <1>       mov    si, ax
8391                                    <1>       ;; esi = src = car * 8
8392                                    <1>       ; esi = src = car * [CHAR_HEIGHT]
8393                                    <1>       ; i = 0
8394                                    <1>       ;add   esi, vgafont8 ; fdata [src+i]
8395                                    <1>       ; 08/08/2016
8396 00002D51 A1[9A950100]              <1>       mov    eax, [VGA_INT43H]
8397 00002D56 09C0                      <1>       or     eax, eax ; 0 ?
8398 00002D58 743F                      <1>       jz     short wfxl_7 ; yes, default font
8399                                    <1>       ;cmp   eax, vgafont16
8400                                    <1>         ;je  short wgfxl_0
8401                                    <1>       ;cmp   eax, vgafont14
8402                                    <1>       ;je    short wgfxl_0
8403                                    <1>       ;cmp   eax, vgafont8
8404                                    <1>       ;je    short wgfxl_0
8405                                    <1>       ;; 05/01/2021 (TRDOS 386 v2.0.3)
8406                                    <1>       ;; user font
8407                                    <1>       ;mov   eax, VGAFONTUSR ; 8x16 or 8x8 or 8x14 font
8408                                    <1>       ;                      ; (256 characters)
8409                                    <1> wgfxl_0:
8410 00002D5A 01C6                      <1>       add    esi, eax
8411                                    <1> wgfxl_1:
8412 00002D5C 28FF                      <1>       sub    bh, bh ; i = 0
8413                                    <1> wgfxl_2:
8414                                    <1>       ; for(i=0;i<8;i++)
8415 00002D5E 57                        <1>       push   edi ; addr
8416 00002D5F 0FB605[9C6E0000]          <1>       movzx  eax, byte [CRT_COLS] ; nbcols
8417 00002D66 F6E7                      <1>       mul    bh ; nbcols*i
8418 00002D68 66C1E003                  <1>       shl    ax, 3 ; i*nbcols*8
8419                                    <1>       ; dest=addr+i*nbcols*8;
8420 00002D6C 01C7                      <1>       add    edi, eax ; dest + j ; j = 0
8421 00002D6E B180                      <1>       mov    cl, 80h ; mask = 0x80;
8422                                    <1>       ; esi = fdata + src + i
8423                                    <1>       ; for(j=0;j<8;j++)
8424 00002D70 29D2                      <1>       sub    edx, edx ; j = 0
8425                                    <1> wgfxl_3:
8426 00002D72 8A06                      <1>       mov    al, [esi] ; al = fdata[src+i]
8427 00002D74 20C8                      <1>       and    al, cl ; if (fdata[src+i] & mask)
8428 00002D76 7402                      <1>       jz     short wgfxl_4  ; data = 0, zf = 1
8429 00002D78 88D8                      <1>       mov    al, bl ; data = attr;
8430                                    <1> wgfxl_4:
8431                                    <1>       ; write_byte(0xa000,dest+j,data);
8432 00002D7A AA                        <1>       stosb  ; dest + j (+ 0A0000h)
8433                                    <1>       ;inc   dl ; j++
8434                                    <1>       ;cmp   dl, 8
8435 00002D7B 80FA07                    <1>       cmp    dl, 7
8436 00002D7E 720E                      <1>       jb     short wgfxl_5
8437 00002D80 5F                        <1>       pop    edi
8438                                    <1>       ; 08/08/2016
8439                                    <1>       ;cmp   bh, 7
8440                                    <1>       ;jnb   short wgfxl_6
8441 00002D81 FEC7                      <1>       inc    bh ; i++
8442 00002D83 3A3D[9E6E0000]            <1>       cmp    bh, [CHAR_HEIGHT]
8443 00002D89 7309                      <1>       jnb    short wgfxl_6
8444 00002D8B 46                        <1>       inc    esi
8445 00002D8C EBD0                      <1>       jmp    short wgfxl_2
8446                                    <1> wgfxl_5:
8447 00002D8E D0E9                      <1>       shr    cl, 1 ; mask >>= 1;
8448 00002D90 FEC2                      <1>       inc    dl ; j++
8449 00002D92 EBDE                      <1>         jmp     short wgfxl_3
8450                                    <1> wgfxl_6:
8451 00002D94 58                        <1>       pop    eax
8452 00002D95 5A                        <1>       pop    edx
8453 00002D96 5B                        <1>       pop    ebx
8454 00002D97 59                        <1>       pop    ecx
8455 00002D98 C3                        <1>       retn
8456                                    <1> wfxl_7:
8457                                    <1>       ; 08/01/2021
8458                                    <1>       ; 05/01/2021
```

```
                                <1>       ; Default font (8x8 or 8x14 or 8x16)
00002D99 A0[9E6E0000]           <1>       mov   al, [CHAR_HEIGHT]
00002D9E 3C08                   <1>       cmp   al, 8
00002DA0 7507                   <1>       jne   short wfxl_8
00002DA2 B8[845F0100]           <1>       mov   eax, vgafont8
00002DA7 EBB1                   <1>       jmp   short wgfxl_0
                                <1> wfxl_8:
00002DA9 3C0E                   <1>       cmp   al, 14
00002DAB 7507                   <1>       jne   short wfxl_9
00002DAD B8[84670100]           <1>       mov   eax, vgafont14
00002DB2 EBA6                   <1>       jmp   short wgfxl_0
                                <1> wfxl_9:
00002DB4 B8[84750100]           <1>       mov   eax, vgafont16
00002DB9 EB9F                   <1>       jmp   short wgfxl_0
                                <1>
                                <1> write_gfx_char_pl4:
                                <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>
00002DBB 51                     <1>       push  ecx
00002DBC 53                     <1>       push  ebx
00002DBD 52                     <1>       push  edx
00002DBE 50                     <1>       push  eax
                                <1> wgfxpl_f0:
                                <1>       ; switch(cheight)
00002DBF 8A25[9E6E0000]         <1>       mov   ah, [CHAR_HEIGHT]
00002DC5 80FC10                 <1>       cmp   ah, 16 ; case 16:
00002DC8 7507                   <1>       jne   short wgfxpl_f1
                                <1>       ; fdata = &vgafont16;
00002DCA BE[84750100]           <1>       mov   esi, vgafont16
00002DCF EB13                   <1>       jmp   short wgfxpl_f3
                                <1> wgfxpl_f1:
00002DD1 80FC0E                 <1>       cmp   ah, 14 ; case 14:
00002DD4 7507                   <1>       jne   short wgfxpl_f2
00002DD6 BE[84670100]           <1>       mov   esi, vgafont14
00002DDB EB07                   <1>       jmp   short wgfxpl_f3
                                <1> wgfxpl_f2:
                                <1>       ; default:
                                <1>       ;  fdata = &vgafont8;
00002DDD BE[845F0100]           <1>       mov   esi, vgafont8
00002DE2 B408                   <1>       mov   ah, 8
                                <1> wgfxpl_f3:
                                <1>       ; al = car
00002DE4 F6E4                   <1>       mul   ah ; ah = cheight
00002DE6 25FFFF0000             <1>       and   eax, 0FFFFh ; car * cheight
                                <1>       ; src = car * cheight;
00002DEB 01C6                   <1>       add   esi, eax ; esi = fdata[src+i]
                                <1>       ; addr=xcurs*8+ycurs*nbcols*64;
00002DED 88F0                   <1>       mov   al, dh ; ycurs
00002DEF 8A25[9C6E0000]         <1>       mov   ah, [CRT_COLS] ; nbcols
00002DF5 F6E4                   <1>       mul   ah
                                <1>       ; 08/08/2016
                                <1>       ;shl   ax, 6 ; * 64
00002DF7 66C1E003               <1>       shl   ax, 3 ; * 8
                                <1>       ;sub   dh, dh ; 0
                                <1>       ;shl   dx, 3 ; xcurs * 8
                                <1>       ;movzx edi, dx
00002DFB 0FB6FA                 <1>       movzx  edi, dl
00002DFE 66C1E703               <1>       shl   di, 3 ; xcurs * 8
00002E02 30F6                   <1>       xor   dh, dh
00002E04 8A15[9E6E0000]         <1>       mov   dl, [CHAR_HEIGHT]
00002E0A 66F7E2                 <1>       mul   dx
                                <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
00002E0D 01C7                   <1>       add   edi, eax ; addr
00002E0F 81C700000A00           <1>       add   edi, 0A0000h
                                <1>       ;
                                <1>       ; outw(VGAREG_SEQU_ADDRESS, 0x0f02);
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
00002E15 66BAC403               <1>       mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
00002E19 66B8020F               <1>       mov   ax, 0F02h
00002E1D 66EF                   <1>       out   dx, ax
00002E1F 66BACE03               <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E23 66B80502               <1>       mov   ax, 0205h
00002E27 66EF                   <1>       out   dx, ax
                                <1>       ;
00002E29 66BACE03               <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E2D F6C380                 <1>       test  bl, 80h ; if(attr&0x80)
00002E30 7406                   <1>       jz    short wgfxpl_f4 ; else
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x1803);
00002E32 66B80318               <1>       mov   ax, 1803h
00002E36 EB04                   <1>       jmp   short wgfxpl_f5
                                <1> wgfxpl_f4:
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0003);
00002E38 66B80300               <1>       mov   ax, 0003h
                                <1> wgfxpl_f5:
00002E3C 66EF                   <1>       out   dx, ax
                                <1>       ;
00002E3E 28FF                   <1>       sub   bh, bh ; i = 0
                                <1> wgfxpl_0:
                                <1>       ; for(i=0;i<cheight;i++)
00002E40 57                     <1>       push  edi ; addr
00002E41 0FB605[9C6E0000]       <1>       movzx eax, byte [CRT_COLS] ; nbcols
00002E48 F6E7                   <1>       mul   bh ; nbcols*i
```

```
8564                                    <1>       ; dest=addr+i*nbcols
8565 00002E4A 01C7                      <1>       add    edi, eax ; dest
8566 00002E4C B580                      <1>       mov    ch, 80h ; mask = 0x80;
8567                                    <1>       ; for(j=0;j<8;j++)
8568 00002E4E 28C9                      <1>       sub    cl, cl ; j = 0
8569                                    <1> wgfxpl_1:
8570 00002E50 D2ED                      <1>       shr    ch, cl ; mask=0x80>>j;
8571                                    <1>       ;
8572                                    <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
8573                                    <1>       ; read_byte(0xa000,dest);
8574                                    <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8575 00002E52 88EC                      <1>       mov    ah, ch
8576 00002E54 B008                      <1>       mov    al, 8
8577 00002E56 66EF                      <1>       out    dx, ax
8578 00002E58 8A07                      <1>       mov    al, [edi] ; ? (io delay?)
8579                                    <1>       ;
8580 00002E5A 28C0                      <1>       sub    al, al ; attr = 0
8581                                    <1>       ; if (fdata[src+i] & mask)
8582 00002E5C 842E                      <1>       test   byte [esi], ch
8583 00002E5E 7404                      <1>       jz     short wgfxpl_2  ; zf = 1
8584                                    <1>       ; write_byte(0xa000,dest,attr&0x0f);
8585 00002E60 88D8                      <1>       mov    al, bl ; attr;
8586 00002E62 240F                      <1>       and    al, 0Fh      ; attr&0x0f
8587                                    <1> wgfxpl_2:
8588                                    <1>       ; write_byte(0xa000,dest,0x00);
8589 00002E64 8807                      <1>       mov    [edi], al ; dest (+ 0A0000h)
8590 00002E66 FEC1                      <1>       inc    cl ; j++
8591 00002E68 80F908                    <1>       cmp    cl, 8
8592 00002E6B 72E3                      <1>       jb     short wgfxpl_1
8593 00002E6D 5F                        <1>       pop    edi
8594                                    <1>       ; 08/08/2016
8595                                    <1>       ;cmp   bh, 7
8596                                    <1>       ;jnb   short wgfxpl_3
8597 00002E6E FEC7                      <1>       inc    bh ; i++
8598 00002E70 3A3D[9E6E0000]            <1>       cmp    bh, [CHAR_HEIGHT]
8599 00002E76 7303                      <1>       jnb    short wgfxpl_3
8600 00002E78 46                        <1>       inc    esi
8601 00002E79 EBC5                      <1>       jmp    short wgfxpl_0
8602                                    <1> wgfxpl_3:
8603                                    <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8604 00002E7B 66B808FF                  <1>       mov    ax, 0FF08h
8605 00002E7F 66EF                      <1>       out    dx, ax
8606 00002E81 66B80500                  <1>       mov    ax, 0005h
8607 00002E85 66EF                      <1>       out    dx, ax
8608 00002E87 66B80300                  <1>       mov    ax, 0003h
8609 00002E8B 66EF                      <1>       out    dx, ax
8610                                    <1>       ;
8611 00002E8D 58                        <1>       pop    eax
8612 00002E8E 5A                        <1>       pop    edx
8613 00002E8F 5B                        <1>       pop    ebx
8614 00002E90 59                        <1>       pop    ecx
8615 00002E91 C3                        <1>       retn
8616                                    <1>
8617                                    <1> vga_write_pixel:
8618                                    <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8619                                    <1>       ;
8620                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
8621                                    <1>       ; vgabios-0.7a (2011)
8622                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
8623                                    <1>       ; 'vgabios.c', 'biosfn_write_pixel'
8624                                    <1>
8625                                    <1>       ; INPUT ->
8626                                    <1>       ;      DX = row (0-239)
8627                                    <1>       ;      CX = column (0-799)
8628                                    <1>       ;      AL = pixel value
8629                                    <1>       ;      (AH = [CRT_MODE])
8630                                    <1>       ; OUTPUT ->
8631                                    <1>       ;      none
8632                                    <1>
8633 00002E92 88C3                      <1>       mov    bl, al ; pixel value
8634                                    <1>       ;mov   ah, [CRT_MODE]
8635 00002E94 BE[B66E0000]              <1>       mov    esi, vga_modes
8636 00002E99 89F7                      <1>       mov    edi, esi
8637 00002E9B 83C710                    <1>       add    edi, vga_mode_count
8638                                    <1> vga_wp_0:
8639 00002E9E AC                        <1>       lodsb
8640 00002E9F 38E0                      <1>       cmp    al, ah ; [CRT_MODE]
8641 00002EA1 7405                      <1>       je     short vga_wp_1
8642 00002EA3 39FE                      <1>       cmp    esi, edi
8643 00002EA5 72F7                      <1>       jb     short vga_wp_0
8644 00002EA7 C3                        <1>       retn   ; nothing to do
8645                                    <1> vga_wp_1:
8646 00002EA8 83C64F                    <1>       add    esi, vga_memmodel - (vga_modes + 1)
8647                                    <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8648 00002EAB BF00000A00                <1>       mov    edi, 0A0000h
8649                                    <1>       ;
8650 00002EB0 803E04                    <1>       cmp    byte [esi], PLANAR4
8651 00002EB3 741D                      <1>       je     short vga_wp_planar
8652 00002EB5 803E03                    <1>       cmp    byte [esi], PLANAR1
8653 00002EB8 7418                      <1>       je     short vga_wp_planar
8654                                    <1> vga_wp_linear8:
8655                                    <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8656 00002EBA 0FB605[9C6E0000]          <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8657 00002EC1 66C1E003                  <1>       shl    ax, 3 ; * 8
8658 00002EC5 66F7E2                    <1>       mul    dx
8659 00002EC8 50                        <1>       push   eax
8660                                    <1>       ;mov   edi, 0A0000h
8661 00002EC9 6601CF                    <1>       add    di, cx
8662 00002ECC 58                        <1>       pop    eax
8663 00002ECD 01C7                      <1>       add    edi, eax ; addr
8664                                    <1>       ; write_byte(0xa000,addr,AL);
8665 00002ECF 881F                      <1>       mov    [edi], bl
8666 00002ED1 C3                        <1>       retn
8667                                    <1> vga_wp_planar:
8668                                    <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
```

```
8669 00002ED2 0FB7C1              <1>       movzx  eax, cx
8670 00002ED5 66C1E803            <1>       shr    ax, 3 ; CX/8
8671 00002ED9 50                  <1>       push   eax
8672 00002EDA 28E4                <1>       sub    ah, ah ; 0
8673 00002EDC A0[9C6E0000]        <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8674 00002EE1 66F7E2              <1>       mul    dx
8675                              <1>       ;mov   edi, 0A0000h
8676 00002EE4 6601C7              <1>       add   di, ax
8677 00002EE7 58                  <1>       pop    eax
8678 00002EE8 01C7                <1>       add    edi, eax ; addr
8679 00002EEA 80E107              <1>       and    cl, 7
8680 00002EED B580                <1>       mov    ch, 80h ; mask
8681 00002EEF D2ED                <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
8682                              <1>
8683                              <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
8684 00002EF1 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8685 00002EF5 88EC                <1>       mov    ah, ch
8686 00002EF7 B008                <1>       mov    al, 8
8687 00002EF9 66EF                <1>       out    dx, ax
8688                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
8689 00002EFB 66B80502            <1>       mov    ax, 0205h
8690 00002EFF 66EF                <1>       out    dx, ax
8691                              <1>       ; data = read_byte(0xa000,addr);
8692 00002F01 8A07                <1>       mov    al, [edi] ; (delay?)
8693                              <1>       ; if (AL & 0x80)
8694                              <1>       ; {
8695                              <1>       ;  outw(VGAREG_GRDC_ADDRESS, 0x1803);
8696                              <1>       ; }
8697 00002F03 F6C380              <1>       test   bl, 80h
8698 00002F06 7406                <1>       jz     short vga_wp_2
8699 00002F08 66B80318            <1>       mov    ax, 1803h
8700 00002F0C 66EF                <1>       out    dx, ax
8701                              <1> vga_wp_2:
8702                              <1>       ; write_byte(0xa000,addr,AL);
8703 00002F0E 881F                <1>       mov    [edi], bl
8704                              <1>       ;
8705                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8706 00002F10 66B808FF            <1>       mov    ax, 0FF08h
8707 00002F14 66EF                <1>       out    dx, ax
8708 00002F16 66B80500            <1>       mov    ax, 0005h
8709 00002F1A 66EF                <1>       out    dx, ax
8710 00002F1C 66B80300            <1>       mov    ax, 0003h
8711 00002F20 66EF                <1>       out    dx, ax
8712                              <1>       ;
8713 00002F22 C3                  <1>       retn
8714                              <1>
8715                              <1> vga_read_pixel:
8716                              <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8717                              <1>       ;
8718                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8719                              <1>       ; vgabios-0.7a (2011)
8720                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8721                              <1>       ; 'vgabios.c', 'biosfn_read_pixel'
8722                              <1>
8723                              <1>       ; INPUT ->
8724                              <1>       ;      DX = row (0-239)
8725                              <1>       ;      CX = column (0-799)
8726                              <1>       ;      (AH = [CRT_MODE])
8727                              <1>       ; OUTPUT ->
8728                              <1>       ;      AL = pixel value
8729                              <1>
8730                              <1>       ;mov   ah, [CRT_MODE]
8731 00002F23 BE[B66E0000]        <1>       mov    esi, vga_modes
8732 00002F28 89F7                <1>       mov    edi, esi
8733 00002F2A 83C710              <1>       add    edi, vga_mode_count
8734                              <1> vga_rp_0:
8735 00002F2D AC                  <1>       lodsb
8736 00002F2E 38E0                <1>       cmp    al, ah ; [CRT_MODE]
8737 00002F30 7405                <1>       je     short vga_rp_1
8738 00002F32 39FE                <1>       cmp    esi, edi
8739 00002F34 72F7                <1>       jb     short vga_rp_0
8740 00002F36 C3                  <1>       retn   ; nothing to do
8741                              <1> vga_rp_1:
8742 00002F37 83C64F              <1>       add    esi, vga_memmodel - (vga_modes + 1)
8743                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8744 00002F3A BF00000A00          <1>       mov    edi, 0A0000h
8745                              <1>       ;
8746 00002F3F 803E04              <1>       cmp    byte [esi], PLANAR4
8747 00002F42 741D                <1>       je     short vga_rp_planar
8748 00002F44 803E03              <1>       cmp    byte [esi], PLANAR1
8749 00002F47 7418                <1>       je     short vga_rp_planar
8750                              <1> vga_rp_linear8:
8751                              <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8752 00002F49 0FB605[9C6E0000]    <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8753 00002F50 66C1E003            <1>       shl    ax, 3 ; * 8
8754 00002F54 66F7E2              <1>       mul    dx
8755 00002F57 50                  <1>       push   eax
8756                              <1>       ;mov   edi, 0A0000h
8757 00002F58 6601CF              <1>       add    di, cx
8758 00002F5B 58                  <1>       pop    eax
8759 00002F5C 01C7                <1>       add    edi, eax ; addr
8760                              <1>       ; attr=read_byte(0xa000,addr);
8761 00002F5E 8A07                <1>       mov    al, [edi] ; pixel value
8762 00002F60 C3                  <1>       retn
8763                              <1> vga_rp_planar:
8764                              <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
8765 00002F61 0FB7C1              <1>       movzx  eax, cx
8766 00002F64 66C1E803            <1>       shr    ax, 3 ; CX/8
8767 00002F68 50                  <1>       push   eax
8768 00002F69 28E4                <1>       sub    ah, ah ; 0
8769 00002F6B A0[9C6E0000]        <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8770 00002F70 66F7E2              <1>       mul    dx
8771                              <1>       ;mov   edi, 0A0000h
8772 00002F73 6601C7              <1>         add     di, ax
8773 00002F76 58                  <1>       pop    eax
```

```
8774 00002F77 01C7              <1>       add    edi, eax ; addr
8775 00002F79 80E107            <1>       and    cl, 7
8776 00002F7C B580              <1>       mov    ch, 80h ; mask
8777 00002F7E D2ED              <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
8778                            <1>       ; attr = 0x00;
8779 00002F80 30DB              <1>       xor    bl, bl ; attr = bl = 0,
8780 00002F82 30C9              <1>       xor    cl, cl ; i = cl = 0
8781                            <1>       ; for(i=0;i<4;i++)
8782                            <1>             ; {
8783                            <1>             ;  outw(VGAREG_GRDC_ADDRESS, (i << 8) | 0x04);
8784                            <1>             ;  data = read_byte(0xa000,addr) & mask;
8785                            <1>             ;  if (data > 0) attr |= (0x01 << i);
8786                            <1>             ; }
8787                            <1> vga_rp_2:
8788 00002F84 88CC              <1>       mov    ah, cl ; i << 8
8789 00002F86 B004              <1>       mov    al, 4  ; | 0x04
8790 00002F88 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8791 00002F8C 66EF              <1>       out    dx, ax
8792                            <1>       ; data = read_byte(0xa000,addr) & mask;
8793 00002F8E 8A07              <1>       mov    al, [edi]
8794 00002F90 20E8              <1>       and    al, ch ; & mask
8795                            <1>       ; if (data > 0) attr |= (0x01 << i);
8796 00002F92 08C0              <1>       or     al, al
8797 00002F94 7408              <1>       jz     short vga_rp_3 ; al = 0
8798 00002F96 B701              <1>       mov    bh, 1
8799 00002F98 D2E7              <1>       shl    bh, cl ; (0x01 << i)
8800 00002F9A 08FB              <1>       or     bl, bh ; attr |= (0x01 << i)
8801 00002F9C 88D8              <1>       mov    al, bl ; pixel value
8802                            <1> vga_rp_3:
8803 00002F9E C3                <1>       retn
8804                            <1>
8805                            <1> vga_beeper:
8806                            <1>       ; 04/08/2016  (TRDOS 386 = TRDOS v2.0)
8807 00002F9F FB                <1>       sti
8808                            <1>       ;mov   bh, [ACTIVE_PAGE]
8809 00002FA0 E9D0F3FFFF        <1>         jmp     beeper_gfx
8810                            <1>
8811                            <1> vga_write_teletype:
8812                            <1>       ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
8813                            <1>       ; 09/12/2017
8814                            <1>       ; 06/08/2016
8815                            <1>       ; 04/08/2016
8816                            <1>       ; 01/08/2016
8817                            <1>       ; 31/07/2016
8818                            <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8819                            <1>       ;
8820                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8821                            <1>       ; vgabios-0.7a (2011)
8822                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8823                            <1>       ; 'vgabios.c', 'biosfn_write_teletype'
8824                            <1>       ; 'biosfn_write_char_only'
8825                            <1>
8826                            <1>       ; INPUT ->
8827                            <1>       ; [CRT_MODE] = current video mode (>7)
8828                            <1>       ; AL = Character to write
8829                            <1>       ; BL = Color of character
8830                            <1>       ; OUTPUT ->
8831                            <1>       ; Regen buffer updated
8832                            <1>
8833                            <1>       ; biosfn_write_teletype (car, page, attr, flag)
8834                            <1>       ; car = character (AL)
8835                            <1>       ; page = 0
8836                            <1>       ; attr = color (BL)
8837                            <1>       ; 'flag' not used
8838                            <1>
8839 00002FA5 8A25[9A6E0000]   <1>       mov    ah, [CRT_MODE]
8840 00002FAB 88C7              <1>       mov    bh, al ; character
8841 00002FAD 668B15[0E890100]  <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
8842                            <1>
8843 00002FB4 BE[BE6E0000]      <1>       mov    esi, vga_g_modes
8844 00002FB9 89F7              <1>       mov    edi, esi
8845 00002FBB 83C708            <1>       add    edi, vga_g_mode_count
8846                            <1> vga_wtty_0:
8847 00002FBE AC                <1>       lodsb
8848 00002FBF 38E0              <1>       cmp    al, ah ; [CRT_MODE]
8849 00002FC1 7405              <1>       je     short vga_wtty_2
8850 00002FC3 39FE              <1>       cmp    esi, edi
8851 00002FC5 72F7              <1>       jb     short vga_wtty_0
8852                            <1> vga_wtty_1:
8853 00002FC7 C3                <1>       retn   ; nothing to do
8854                            <1> vga_wtty_2:
8855 00002FC8 80FF07            <1>       cmp    bh, 07h ; bell (beep)
8856 00002FCB 74D2              <1>       je     short vga_beeper  ; u11
8857 00002FCD 80FF08            <1>       cmp    bh, 08h ; backspace
8858 00002FD0 7508              <1>       jne    short vga_wtty_3
8859                            <1>       ; if(xcurs>0)xcurs--;
8860 00002FD2 08D2              <1>       or     dl, dl ; xcurs (column)
8861 00002FD4 74F1              <1>       jz     short vga_wtty_1
8862 00002FD6 FECA              <1>       dec    dl ; xcurs--;
8863 00002FD8 EB55              <1>       jmp   short vga_wtty_12
8864                            <1> vga_wtty_3:
8865 00002FDA 80FF0D            <1>       cmp    bh, 0Dh ; carriage return (\r)
8866 00002FDD 7504              <1>       jne    short vga_wtty_4
8867                            <1>       ; xcurs=0;
8868 00002FDF 28D2              <1>       sub    dl, dl ; 0
8869 00002FE1 EB4C              <1>       jmp   short vga_wtty_12
8870                            <1> vga_wtty_4:
8871 00002FE3 80FF0A            <1>       cmp    bh, 0Ah ; new line (\n)
8872 00002FE6 7504              <1>       jne    short vga_wtty_5
8873                            <1>       ; ycurs++;
8874 00002FE8 FEC6              <1>       inc    dh ; next row
8875 00002FEA EB5E              <1>       jmp   short vga_wtty_11
8876                            <1> vga_wtty_5:
8877 00002FEC 80FF09            <1>       cmp    bh, 09h ; tab stop
8878 00002FEF 7523              <1>       jne    short vga_wtty_8
```

```
8879 00002FF1 88D0                <1>       mov    al, dl
8880                              <1>       ;cbw
8881 00002FF3 30E4                <1>       xor    ah, ah ; 09/12/2017
8882 00002FF5 B108                <1>       mov    cl, 8
8883 00002FF7 F6F1                <1>       div    cl
8884 00002FF9 28E1                <1>       sub    cl, ah
8885                              <1>       ;
8886 00002FFB B720                <1>       mov    bh, 20h ; space
8887                              <1> vga_wtty_6: ; tab stop loop
8888                              <1>       ;push  cx
8889                              <1>       ;push  bx
8890                              <1>       ; 12/04/2021
8891 00002FFD 51                  <1>       push   ecx
8892 00002FFE 53                  <1>       push   ebx
8893 00002FFF E810000000          <1>       call   vga_wtty_8
8894                              <1>       ;pop   bx  ; bh = character, bl = color
8895                              <1>       ;pop   cx
8896                              <1>       ; 12/04/2021
8897 00003004 5B                  <1>       pop    ebx  ; bh = character, bl = color
8898 00003005 59                  <1>       pop    ecx
8899 00003006 FEC9                <1>       dec    cl
8900 00003008 7409                <1>       jz     short vga_wtty_7
8901 0000300A 668B15[0E890100]    <1>       mov    dx, [CURSOR_POSN] ; new cursor position (pg 0)
8902 00003011 EBEA                <1>       jmp    short vga_wtty_6
8903                              <1> vga_wtty_7:
8904 00003013 C3                  <1>       retn
8905                              <1>       ;
8906                              <1> vga_wtty_8:
8907 00003014 83C64F              <1>       add    esi, vga_g_memmodel - (vga_g_modes + 1)
8908                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8909 00003017 BF00000A00          <1>       mov    edi, 0A0000h
8910                              <1>       ;
8911 0000301C 88F8                <1>       mov    al, bh ; character
8912                              <1>       ;
8913 0000301E 803E04              <1>       cmp    byte [esi], PLANAR4
8914 00003021 7414                <1>       je     short vga_wtty_planar
8915 00003023 803E03              <1>       cmp    byte [esi], PLANAR1
8916 00003026 740F                <1>       je     short vga_wtty_planar
8917                              <1> vga_wtty_linear8:
8918                              <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
8919                              <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
8920                              <1>       ; [CRT_COLS] = nbcols
8921 00003028 E8EBFCFFFF          <1>       call   write_gfx_char_lin
8922 0000302D EB0D                <1>       jmp    short vga_wtty_9
8923                              <1>
8924                              <1> vga_wtty_12:
8925                              <1>       ; 09/07/2016
8926                              <1>       ; set cursor position
8927                              <1>       ; NOTE: Hardware cursor position will not be set
8928                              <1>       ;   in any VGA modes (>7)
8929                              <1>       ;   But, cursor position will be saved into
8930                              <1>       ;   [CURSOR_POSN].
8931                              <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
8932                              <1>       ;   (page 0) for all graphics modes.
8933                              <1>
8934 0000302F 668915[0E890100]    <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
8935                              <1>       ; 04/08/2016
8936                              <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
8937                              <1>       ;call  _set_cpos
8938 00003036 C3                  <1>       retn
8939                              <1>
8940                              <1> vga_wtty_planar:
8941                              <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
8942                              <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
8943                              <1>       ; [CRT_COLS]= nbcols, [CHAR_HEIGHT] = cheight
8944 00003037 E87FFDFFFF          <1>       call   write_gfx_char_pl4
8945                              <1> vga_wtty_9:
8946 0000303C FEC2                <1>       inc    dl ; xcurs++;
8947                              <1> vga_wtty_10:
8948                              <1>       ; Do we need to wrap ?
8949                              <1>       ; if(xcurs==nbcols)
8950 0000303E 3A15[9C6E0000]      <1>       cmp    dl, [CRT_COLS] ; [VGA_COLS]
8951 00003044 7204                <1>       jb     short vga_wtty_11 ; no
8952 00003046 28D2                <1>       sub    dl, dl ; xcurs=0;
8953 00003048 FEC6                <1>       inc    dh ;  ycurs++;
8954                              <1> vga_wtty_11:
8955                              <1>       ; Do we need to scroll ?
8956                              <1>       ; if(ycurs==nbrows)
8957 0000304A 3A35[A26E0000]      <1>       cmp    dh, [VGA_ROWS]
8958 00003050 72DD                <1>       jb     short vga_wtty_12 ; no
8959                              <1>       ;
8960                              <1>       ; biosfn_scroll (nblines,attr,rul,cul,rlr,clr,page,dir)
8961                              <1>       ; al = nblines = 1, bl = attr (color) = 0
8962                              <1>       ; ch = rul, cl = cul, dh = rlr, dl = clr, page = 0
8963                              <1>       ; dir = SCROLL_UP
8964                              <1>
8965 00003052 B001                <1>       mov    al, 1
8966 00003054 28DB                <1>       sub    bl, bl ; 0 ; blank/black line (attr=0) will be used
8967 00003056 6629C9              <1>       sub    cx, cx ; 0,0
8968                              <1>
8969                              <1>       ; 06/08/2016
8970 00003059 8A35[A26E0000]      <1>       mov    dh, [VGA_ROWS]
8971 0000305F FECE                <1>       dec    dh ; nbrows -1
8972                              <1>
8973                              <1>       ;push  dx ; 04/08/2016
8974                              <1>       ; 12/04/2021
8975 00003061 52                  <1>       push   edx
8976 00003062 8A15[9C6E0000]      <1>       mov    dl, [CRT_COLS]
8977 00003068 FECA                <1>       dec    dl ; nbcols -1
8978                              <1>
8979 0000306A 8A25[9A6E0000]      <1>       mov    ah, [CRT_MODE]
8980                              <1>
8981                              <1>       ; biosfn_scroll(0x01,0x00,0,0,nbrows-1,nbcols-1,page,SCROLL_UP);
8982 00003070 E810F5FFFF          <1>       call   vga_graphics_up
8983                              <1>       ; 04/08/2016
```

```
                                <1>       ;pop   dx
                                <1>       ; 12/04/2021
00003075 5A                     <1>       pop    edx
                                <1>
                                <1>       ;dec   dh ; ycurs-=1
00003076 EBB7                   <1>       jmp    short vga_wtty_12
                                <1>
                                <1> font_setup:
                                <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; 09/07/2016
                                <1>       ; character generator (font loading) functions
                                <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', 'int10_func'
                                <1>
                                <1>       ; AX = 1100H ; Load User-Defined Font (EGA/VGA)
                                <1>       ;
                                <1>         ; BH    height of each character (bytes per character definition)
                                <1>         ; (BL   font block to load (EGA: 0-3; VGA: 0-7))
                                <1>       ; CX    number of characters to redefine (<=256)
                                <1>         ; DX    ASCII code of the first character defined at ES:BP
                                <1>         ; EBP      address of font-definition information
                                <1>       ;      (in user's memory space)
                                <1>
                                <1>       ; case 0x11:
                                <1>       ; switch(GET_AL())
                                <1>             ; {
                                <1>       ; case 0x00:
                                <1>         ; case 0x10:
                                <1>         ; biosfn_load_text_user_pat(GET_AL(),ES,BP,CX,DX,GET_BL(),GET_BH());
                                <1>         ; break;
                                <1>
                                <1>       ; AX = 1110H ; Load and Activate User-Defined Font (EGA/VGA)
00003078 08C0                   <1>       or     al, al ; 0
0000307A 7404                   <1>       jz     short font_setup_0
0000307C 3C10                   <1>       cmp    al, 10h
0000307E 7511                   <1>       jne    short font_setup_1
                                <1> font_setup_0:
00003080 E8CE000000             <1>       call   transfer_user_fonts
00003085 721C                   <1>       jc     short font_setup_error
00003087 E8B2010000             <1>       call   load_text_user_pat
0000308C E960EAFFFF             <1>       jmp    VIDEO_RETURN
                                <1> font_setup_1:
                                <1>       ; AX = 1101H ; Load ROM 8x14 Character Set (EGA/VGA)
                                <1>       ; case 0x01:
                                <1>       ; case 0x11:
                                <1>         ; biosfn_load_text_8_14_pat(GET_AL(),GET_BL());
                                <1>         ; break;
00003091 3C01                   <1>       cmp    al, 1
00003093 7404                   <1>       je     short font_setup_2
00003095 3C11                   <1>       cmp    al, 11h
00003097 7511                   <1>       jne    short font_setup_3
                                <1> font_setup_2:
                                <1>       ; AX = 1111H ; Load and Activate ROM 8x14 Character Set (EGA/VGA)
                                <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
00003099 E8DC020000             <1>       call   load_text_8_14_pat
0000309E E94EEAFFFF             <1>       jmp    VIDEO_RETURN
                                <1> font_setup_error:
000030A3 29C0                   <1>       sub    eax, eax ; 0 -> fonts could not be loaded
000030A5 E94CEAFFFF             <1>       jmp    _video_return
                                <1> font_setup_3:
                                <1>       ; AX = 1102H ; Load ROM 8x8 Character Set (EGA/VGA)
                                <1>       ; case 0x02:
                                <1>       ; case 0x12:
                                <1>         ; biosfn_load_text_8_8_pat(GET_AL(),GET_BL());
                                <1>         ; break;
000030AA 3C02                   <1>       cmp    al, 2
000030AC 7404                   <1>       je     short font_setup_4
000030AE 3C12                   <1>       cmp    al, 12h
000030B0 750A                   <1>       jne    short font_setup_5
                                <1> font_setup_4:
                                <1>       ; AX = 1112H ; Load and Activate ROM 8x8 Character Set (EGA/VGA)
                                <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
000030B2 E8F3020000             <1>       call   load_text_8_8_pat
000030B7 E935EAFFFF             <1>       jmp    VIDEO_RETURN
                                <1> font_setup_5:
                                <1>       ; AX = 1104H ; Load ROM 8x16 Character Set (EGA/VGA)
                                <1>       ; case 0x04:
                                <1>       ; case 0x14:
                                <1>         ; biosfn_load_text_8_16_pat(GET_AL(),GET_BL());
                                <1>         ; break;
000030BC 3C04                   <1>       cmp    al, 4
000030BE 7404                   <1>       je     short font_setup_6
000030C0 3C14                   <1>       cmp    al, 14h
000030C2 750A                   <1>       jne    short font_setup_7
                                <1> font_setup_6:
                                <1>       ; AX = 1114H ; Load and Activate ROM 8x16 Character Set (EGA/VGA)
                                <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
000030C4 E82A030000             <1>       call   load_text_8_16_pat
000030C9 E923EAFFFF             <1>       jmp    VIDEO_RETURN
                                <1> font_setup_7:
                                <1>       ; Note: AX=1120h (Setup INT 1Fh, EXT_PTR) is not needed
                                <1>       ; for TRDOS 386 (TRDOS v2.0) video functionality;
                                <1>       ; because, originally EXT_PTR (font address) was used for
                                <1>       ; chars 80h to 0FFh (after the first 128 ASCII char fonts), for
                                <1>       ; CGA graphics mode; currenty, 'vgafont8' address has 256 chars!
                                <1>       ;
                                <1>       ; case 0x20:
                                <1>         ; biosfn_load_gfx_8_8_chars(ES,BP);
                                <1>         ; break;
                                <1>       ; case 0x21:
                                <1>         ; biosfn_load_gfx_user_chars(ES,BP,CX,GET_BL(),GET_DL());
                                <1>         ; break;
```

```
                                <1>       ; AX = 1121H ; Setup User-Defined Font for Graphics Mode (VGA)
                                <1>       ; BL   screen rows code: 00H = user-specified (in DL)
                                <1>         ;                        01H = 14 rows
                                <1>         ;                        02H = 25 rows
                                <1>         ;                        03H = 43 rows
                                <1>         ; CX   bytes per character definition
                                <1>         ; DL   (when BL=0) custom number of character rows on screen
                                <1>         ; EBP  address of font-definition information (user's mem space)
                                <1>
000030CE 3C21                   <1>       cmp    al, 21h
000030D0 7531                   <1>       jne    short font_setup_9
                                <1>
                                <1>       ; TRDOS 386 modification !
                                <1>       ; dh = 0 -> 256 characters
                                <1>       ; dh = 80h -> second 128 characters
                                <1>       ; dh = 0FFh -> first 128 characters
                                <1>
                                <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ;push  ebx
000030D2 51                     <1>       push   ecx
000030D3 52                     <1>       push   edx
000030D4 30D2                   <1>       xor    dl, dl
000030D6 88CF                   <1>       mov    bh, cl ; character height
000030D8 66B90001               <1>       mov    cx, 100h ; 256
000030DC 08F6                   <1>       or     dh, dh ; 0
000030DE 7410                   <1>       jz     short  font_setup_8
000030E0 FECD                   <1>       dec    ch ; cx = 0
000030E2 80FEFF                 <1>       cmp    dh, 0FFh
000030E5 7409                   <1>       je     short font_setup_8 ; 1st 128 chars
                                <1>       ; 2nd 128 chars
000030E7 80FE80                 <1>       cmp    dh, 80h
000030EA 75B7                   <1>       jne    short font_setup_error ; invalid !
000030EC 88F1                   <1>       mov    cl, dh
000030EE 86D6                   <1>       xchg   dl, dh
                                <1>       ; number of chars, cx = 80h
                                <1>       ; start char, dl = 80h
                                <1> font_setup_8:
000030F0 E85E000000             <1>       call   transfer_user_fonts
000030F5 5A                     <1>       pop    edx
000030F6 59                     <1>       pop    ecx
                                <1>       ;pop   ebx
000030F7 72AA                   <1>       jc     short font_setup_error
                                <1>       ; ebp = user's font data address in system's memory space
000030F9 E83F030000             <1>       call   load_gfx_user_chars
000030FE E9EEE9FFFF             <1>         jmp     VIDEO_RETURN
                                <1> font_setup_9:
                                <1>       ; case 0x22:
                                <1>         ; biosfn_load_gfx_8_14_chars(GET_BL());
                                <1>         ; break;
00003103 3C22                   <1>       cmp    al, 22h
00003105 750A                   <1>       jne    short font_setup_10
00003107 E86D030000             <1>       call   load_gfx_8_14_chars
0000310C E9E0E9FFFF             <1>       jmp   VIDEO_RETURN
                                <1> font_setup_10:
                                <1>       ; case 0x23:
                                <1>         ; biosfn_load_gfx_8_8_dd_chars(GET_BL());
                                <1>         ; break;
00003111 3C23                   <1>       cmp    al, 23h
00003113 750A                   <1>       jne    short font_setup_11
00003115 E8A0030000             <1>       call   load_gfx_8_8_chars
0000311A E9D2E9FFFF             <1>       jmp   VIDEO_RETURN
                                <1> font_setup_11:
                                <1>       ; case 0x24:
                                <1>         ; biosfn_load_gfx_8_16_chars(GET_BL());
                                <1>         ; break;
0000311F 3C24                   <1>       cmp    al, 24h
00003121 750A                   <1>       jne    short font_setup_12
00003123 E8D3030000             <1>       call   load_gfx_8_16_chars
00003128 E9C4E9FFFF             <1>       jmp   VIDEO_RETURN
                                <1> font_setup_12:
                                <1>       ; case 0x30:
                                <1>         ; biosfn_get_font_info(GET_BH(),&ES,&BP,&CX,&DX);
                                <1>         ; break;
0000312D 3C30                   <1>       cmp    al, 30h
0000312F 750A                   <1>       jne    short font_setup_13
00003131 E806040000             <1>       call   get_font_info
                                <1>       ; eax = return value (info: 4 bytes for 4 parms)
                                <1>       ; eax = 0 -> invalid function (input)
00003136 E9BBE9FFFF             <1>       jmp    _video_return
                                <1> font_setup_13:
0000313B 3C03                   <1>       cmp    al, 03h ; AX = 1103h
0000313D 750D                   <1>       jne    short font_setup_14
                                <1>       ; biosfn_set_text_block_specifier:
                                <1>       ; BL = font block selector code
                                <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
                                <1>       ; (It is as BL = 0 for TRDOS 386)
0000313F 66BAC403               <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
                                <1>       ;mov   ah, bl
00003143 28E4                   <1>       sub    ah, ah ; 0
                                <1>       ;mov   al, 03h
00003145 66EF                   <1>       out    dx, ax
00003147 E9A5E9FFFF             <1>       jmp   VIDEO_RETURN
                                <1>
                                <1> font_setup_14:
0000314C 29C0                   <1>       sub    eax, eax ; 0 = invalid function
0000314E E9A3E9FFFF             <1>       jmp   _video_return
                                <1>
                                <1> transfer_user_fonts:
                                <1>       ; 19/01/2021
                                <1>       ; 09/01/2021
                                <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>
                                <1>       ; BH    height of each character (bytes per character)
                                <1>       ; CX    number of characters to redefine (<=256)
                                <1>       ; DX    ASCII code of the first character defined at EBP
```

```
9194                                    <1>       ; EBP  address of font-definition information
9195                                    <1>       ;      (in user's memory space)
9196                                    <1>
9197                                    <1>       ; Modified registers: eax, edx, ecx, esi, edi, ebp
9198                                    <1>       ;
9199                                    <1>       ; output:
9200                                    <1>       ;      ebp = user font data address in system memory
9201                                    <1>
9202 00003153 81E2FFFF0000              <1>       and    edx, 0FFFFh
9203 00003159 81E1FFFF0000              <1>       and    ecx, 0FFFFh
9204 0000315F 7537                      <1>       jnz    short transfer_user_fonts_5
9205                                    <1>
9206 00003161 09D2                      <1>       or     edx, edx
9207 00003163 7531                      <1>       jnz    short transfer_user_fonts_4
9208 00003165 09ED                      <1>       or     ebp, ebp
9209 00003167 752D                      <1>       jnz    short transfer_user_fonts_4
9210                                    <1>
9211                                    <1>       ; cx = 0, dx = 0, ebp = 0
9212                                    <1>       ; copy system font to user font
9213                                    <1>
9214 00003169 B140                      <1>       mov    cl, 64 ; 64 dwords
9215                                    <1>
9216 0000316B 80FF10                    <1>       cmp    bh, 16
9217 0000316E 7417                      <1>       je     short transfer_user_fonts_2
9218 00003170 80FF08                    <1>       cmp    bh, 8
9219 00003173 7406                      <1>       je     short transfer_user_fonts_1
9220                                    <1>
9221 00003175 BD[84670100]              <1>       mov    ebp, vgafont14
9222 0000317A C3                        <1>       retn
9223                                    <1>
9224                                    <1> transfer_user_fonts_1:
9225 0000317B BF00500900                <1>       mov    edi, VGAFONT8USER
9226 00003180 BE[845F0100]              <1>       mov    esi, vgafont8
9227 00003185 EB0A                      <1>       jmp    short transfer_user_fonts_3
9228                                    <1>
9229                                    <1> transfer_user_fonts_2:
9230 00003187 BF00400900                <1>       mov    edi, VGAFONT16USER
9231 0000318C BE[84750100]              <1>       mov    esi, vgafont16
9232                                    <1> transfer_user_fonts_3:
9233 00003191 89FD                      <1>       mov    ebp, edi
9234 00003193 F3A5                      <1>       rep    movsd
9235 00003195 C3                        <1>       retn
9236                                    <1>
9237                                    <1> transfer_user_fonts_4:
9238 00003196 F9                        <1>       stc
9239 00003197 C3                        <1>       retn
9240                                    <1>
9241                                    <1> transfer_user_fonts_5:
9242 00003198 09ED                      <1>       or     ebp, ebp
9243 0000319A 74FA                      <1>       jz     short transfer_user_fonts_4 ; invalid address !
9244                                    <1>
9245 0000319C 6681F90001                <1>       cmp    cx, 256
9246 000031A1 77F3                      <1>       ja     short transfer_user_fonts_4
9247 000031A3 29D1                      <1>       sub    ecx, edx
9248 000031A5 76EF                      <1>       jna    short transfer_user_fonts_4
9249                                    <1>
9250 000031A7 80FF0E                    <1>       cmp    bh, 14 ; 8x14 font
9251                                    <1>                     ; (there is not an alternative buffer)
9252 000031AA 7525                      <1>       jne    short transfer_user_fonts_6
9253                                    <1>
9254                                    <1>       ; use system's 8x14 font space if permission flag is 1
9255 000031AC F605[7E120300]80          <1>       test   byte [ufont], 80h
9256 000031B3 74E1                      <1>       jz     short transfer_user_fonts_4 ; not allowed
9257                                    <1>
9258                                    <1>       ; permission is given (for vgafont14 location etc.)
9259                                    <1>       ; (for permanent font modification)
9260                                    <1>       ;
9261                                    <1>       ; 19/01/2021
9262                                    <1>       ; Note: Permission flag can be set by 'root' while
9263                                    <1>       ;      system is not in multi tasking/user mode
9264                                    <1>       ;      while [multi_tasking] = 0 and [u.uid] = 0
9265                                    <1>
9266 000031B5 52                        <1>       push   edx
9267 000031B6 30E4                      <1>       xor    ah, ah
9268 000031B8 88F8                      <1>       mov    al, bh ; mov al, 14
9269 000031BA 66F7E1                    <1>       mul    cx
9270                                    <1>              ; ecx = byte count
9271 000031BD 89C1                      <1>       mov    ecx, eax
9272 000031BF 5A                        <1>       pop    edx
9273 000031C0 30E4                      <1>       xor    ah, ah
9274 000031C2 88F8                      <1>       mov    al, bh ; mov ax, 14 ; bytes per character
9275 000031C4 66F7E2                    <1>       mul    dx
9276 000031C7 6689C2                    <1>       mov    dx, ax ; char offset
9277 000031CA BF[84670100]              <1>       mov    edi, vgafont14
9278 000031CF EB4C                      <1>       jmp    short transfer_user_fonts_8
9279                                    <1> transfer_user_fonts_6:
9280 000031D1 80FF08                    <1>       cmp    bh, 8 ; 8x8 font
9281 000031D4 7522                      <1>       jne    short transfer_user_fonts_7 ; 8x16 font
9282 000031D6 66C1E203                  <1>       shl    dx, 3 ; * 8
9283 000031DA 66C1E103                  <1>       shl    cx, 3 ; * 8
9284                                    <1>       ; 09/01/2021
9285 000031DE BF00500900                <1>       mov    edi, VGAFONT8USER
9286 000031E3 F605[7E120300]08          <1>       test   byte [ufont], 8  ; already loaded ?
9287 000031EA 7531                      <1>       jnz    short transfer_user_fonts_8 ; yes
9288 000031EC BE[845F0100]              <1>       mov    esi, vgafont8
9289 000031F1 E83B000000                <1>       call   transfer_user_fonts_10
9290 000031F6 EB25                      <1>       jmp    short transfer_user_fonts_8
9291                                    <1> transfer_user_fonts_7:
9292 000031F8 80FF10                    <1>       cmp    bh, 16 ; 8x16 font
9293 000031FB 7599                      <1>       jne    short transfer_user_fonts_4  ; invalid !
9294 000031FD 66C1E204                  <1>       shl    dx, 4 ; * 16
9295 00003201 66C1E104                  <1>       shl    cx, 4 ; * 16
9296 00003205 BF00400900                <1>       mov    edi, VGAFONT16USER
9297 0000320A F605[7E120300]10          <1>       test   byte [ufont], 16  ; already loaded ?
9298 00003211 750A                      <1>       jnz    short transfer_user_fonts_8 ; yes
```

```
9299 00003213 BE[84750100]        <1>       mov   esi, vgafont16
9300 00003218 E814000000          <1>       call  transfer_user_fonts_10
9301                              <1> transfer_user_fonts_8:
9302 0000321D 01D7                <1>       add   edi, edx ; char offset
9303                              <1>       ; 09/07/2016
9304                              <1>       ;and  ecx, 0FFFFh
9305                              <1>       ; ECX = byte count
9306                              <1>       ;push ecx
9307 0000321F 89EE                <1>       mov   esi, ebp ; user's font buffer
9308                              <1>       ; 09/01/2021
9309 00003221 89FD                <1>       mov   ebp, edi ; system addr for user's font
9310                              <1>       ; 05/01/2021
9311                              <1>       ;mov  edi, Cluster_Buffer  ; system buffer
9312 00003223 E846E80000          <1>       call  transfer_from_user_buffer
9313                              <1>       ;pop  ecx
9314                              <1>       ; ecx = transfer (byte) count = character count
9315 00003228 7206                <1>       jc    short transfer_user_fonts_9
9316                              <1>       ; 05/01/2021
9317                              <1>       ;mov  ebp, Cluster_Buffer
9318                              <1>
9319 0000322A 083D[7E120300]      <1>       or    byte [ufont], bh
9320                              <1>                   ; 8x8 or 8x16 user font ready
9321                              <1> transfer_user_fonts_9:
9322 00003230 C3                  <1>       retn
9323                              <1>
9324                              <1> transfer_user_fonts_10:
9325                              <1>       ; 09/01/2021
9326 00003231 56                  <1>       push  esi
9327 00003232 57                  <1>       push  edi
9328 00003233 51                  <1>       push  ecx
9329 00003234 66B94000            <1>       mov   cx, 64
9330 00003238 F3A5                <1>       rep   movsd
9331 0000323A 59                  <1>       pop   ecx
9332 0000323B 5F                  <1>       pop   edi
9333 0000323C 5E                  <1>       pop   esi
9334 0000323D C3                  <1>       retn
9335                              <1>
9336                              <1> load_text_user_pat:
9337                              <1>       ; 26/07/2016
9338                              <1>       ; 09/07/2016
9339                              <1>       ; load user defined (EGA/VGA) text fonts
9340                              <1>       ;
9341                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
9342                              <1>       ; vgabios-0.7a (2011)
9343                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
9344                              <1>       ; 'vgabios.c', 'biosfn_load_text_user_pat'
9345                              <1>
9346                              <1>       ; biosfn_load_text_user_pat (AL,ES,BP,CX,DX,BL,BH)
9347                              <1>
9348                              <1>       ; get_font_access();
9349                              <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9350                              <1>       ; for(i=0;i<CX;i++)
9351                              <1>       ; {
9352                              <1>       ;  src = BP + i * BH;
9353                              <1>       ;  dest = blockaddr + (DX + i) * 32;
9354                              <1>       ;  memcpyb(0xA000, dest, ES, src, BH);
9355                              <1>       ; }
9356                              <1>       ; release_font_access();
9357                              <1>       ; if(AL>=0x10)
9358                              <1>       ; {
9359                              <1>       ; set_scan_lines(BH);
9360                              <1>       ; }
9361                              <1>
9362 0000323E 50                  <1>       push  eax
9363 0000323F E83C000000          <1>       call  get_font_access
9364 00003244 28DB                <1>       sub   bl, bl ; i = 0
9365                              <1> ltup_1:
9366 00003246 88D8                <1>       mov   al, bl
9367 00003248 F6E7                <1>       mul   bh
9368 0000324A 0FB7F0              <1>       movzx esi, ax
9369 0000324D 01EE                <1>       add   esi, ebp
9370 0000324F 88D8                <1>       mov   al, bl
9371 00003251 28E4                <1>       sub   ah, ah
9372 00003253 6601D0              <1>       add   ax, dx ; (DX + i)
9373 00003256 66C1E005            <1>       shl   ax, 5  ; * 32
9374 0000325A 0FB7F8              <1>       movzx edi, ax
9375 0000325D 81C700000A00        <1>       add   edi, 0A0000h
9376 00003263 51                  <1>       push  ecx
9377 00003264 0FB6CF              <1>       movzx ecx, bh
9378 00003267 F3A4                <1>       rep   movsb
9379 00003269 59                  <1>       pop   ecx
9380 0000326A FEC3                <1>       inc   bl
9381 0000326C 38CB                <1>       cmp   bl, cl
9382 0000326E 75D6                <1>       jne   short ltup_1
9383                              <1>       ;
9384 00003270 E840000000          <1>       call  release_font_access
9385                              <1>       ;
9386 00003275 58                  <1>       pop   eax
9387                              <1>       ; if(AL>=0x10)
9388 00003276 3C10                <1>       cmp   al, 10h
9389 00003278 7205                <1>       jb    short ltup_2
9390                              <1>       ; set_scan_lines(BH);
9391 0000327A E875000000          <1>       call  set_scan_lines
9392                              <1> ltup_2:
9393 0000327F C3                  <1>       retn
9394                              <1>
9395                              <1> get_font_access:
9396                              <1>       ; 09/07/2016
9397                              <1>       ;
9398                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
9399                              <1>       ; vgabios-0.7a (2011)
9400                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
9401                              <1>       ; 'vgabios.c', 'get_font_access'
9402                              <1>
9403                              <1>       ; get_font_access()
```

```
9404 00003280 52                <1>       push   edx
9405 00003281 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
9406 00003285 66B80001          <1>       mov    ax, 0100h
9407 00003289 66EF              <1>       out    dx, ax
9408 0000328B 66B80204          <1>       mov    ax, 0402h
9409 0000328F 66EF              <1>       out    dx, ax
9410 00003291 66B80407          <1>       mov    ax, 0704h
9411 00003295 66EF              <1>       out    dx, ax
9412 00003297 66B80003          <1>       mov    ax, 0300h
9413 0000329B 66EF              <1>       out    dx, ax
9414 0000329D 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
9415 000032A1 66B80402          <1>       mov    ax, 0204h
9416 000032A5 66EF              <1>       out    dx, ax
9417 000032A7 66B80500          <1>       mov    ax, 0005h
9418 000032AB 66EF              <1>       out    dx, ax
9419 000032AD 66B80604          <1>       mov    ax, 0406h
9420 000032B1 66EF              <1>       out    dx, ax
9421 000032B3 5A                <1>       pop    edx
9422 000032B4 C3                <1>       retn
9423                            <1>
9424                            <1> release_font_access:
9425                            <1>       ; 29/07/2016
9426                            <1>       ; 09/07/2016
9427                            <1>       ;
9428                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9429                            <1>       ; vgabios-0.7a (2011)
9430                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9431                            <1>       ; 'vgabios.c', 'release_font_access'
9432                            <1>
9433 000032B5 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
9434 000032B9 66B80001          <1>       mov    ax, 0100h
9435 000032BD 66EF              <1>       out    dx, ax
9436 000032BF 66B80203          <1>       mov    ax, 0302h
9437 000032C3 66EF              <1>       out    dx, ax
9438 000032C5 66B80403          <1>       mov    ax, 0304h
9439 000032C9 66EF              <1>       out    dx, ax
9440 000032CB 66B80003          <1>       mov    ax, 0300h
9441 000032CF 66EF              <1>       out    dx, ax
9442 000032D1 66BACC03          <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
9443 000032D5 EC                <1>       in     al, dx
9444 000032D6 2401              <1>       and    al, 01h
9445 000032D8 C0E002            <1>       shl    al, 2
9446 000032DB 0C0A              <1>       or     al, 0Ah
9447 000032DD 88C4              <1>       mov    ah, al
9448 000032DF B006              <1>       mov    al, 06h
9449 000032E1 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
9450 000032E5 66EF              <1>       out    dx, ax
9451 000032E7 66B80400          <1>       mov    ax, 0004h
9452 000032EB 66EF              <1>       out    dx, ax
9453 000032ED 66B80510          <1>       mov    ax, 1005h
9454 000032F1 66EF              <1>       out    dx, ax
9455 000032F3 C3                <1>       retn
9456                            <1>
9457                            <1> set_scan_lines:
9458                            <1>       ; 09/07/2016
9459                            <1>       ;
9460                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9461                            <1>       ; vgabios-0.7a (2011)
9462                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9463                            <1>       ; 'vgabios.c', 'set_scan_lines'
9464                            <1>
9465                            <1>       ; set_scan_lines(lines)
9466                            <1>       ; BH = lines
9467                            <1>
9468                            <1>       ; outb(crtc_addr, 0x09);
9469 000032F4 66BAD403          <1>       mov    dx, 3D4h ; CRTC_ADDRESS = 3D4h (always)
9470 000032F8 B009              <1>       mov    al, 09h
9471 000032FA EE                <1>       out    dx, al
9472                            <1>       ; crtc_r9 = inb(crtc_addr+1);
9473 000032FB 6642              <1>       inc    dx ; 3D5h
9474 000032FD EC                <1>       in     al, dx
9475                            <1>       ; crtc_r9 = (crtc_r9 & 0xe0) | (lines - 1);
9476 000032FE 24E0              <1>       and    al, 0E0h
9477 00003300 FECF              <1>       dec    bh ; lines - 1
9478 00003302 08F8              <1>       or     al, bh
9479                            <1>       ; outb(crtc_addr+1, crtc_r9);
9480 00003304 EE                <1>       out    dx, al
9481                            <1>       ;inc   bh
9482                            <1>       ; if(lines==8)
9483                            <1>       ;cmp   bh, 8
9484 00003305 80FF07            <1>       cmp    bh, 7
9485 00003308 7506              <1>       jne    short ssl_1
9486                            <1>       ; biosfn_set_cursor_shape(0x06,0x07);
9487 0000330A 66B90706          <1>       mov    cx, 0607h
9488 0000330E EB06              <1>       jmp    short ssl_2
9489                            <1> ssl_1:
9490                            <1>       ; biosfn_set_cursor_shape(lines-4,lines-3);
9491 00003310 88F9              <1>       mov    cl, bh ; lines - 1
9492 00003312 88CD              <1>       mov    ch, cl ; lines - 1 (16 -> 15)
9493 00003314 FECD              <1>       dec    ch  ; lines - 2 (16 -> 14)
9494                            <1> ssl_2:
9495                            <1>       ; CH = start line, CL = stop line
9496 00003316 B40A              <1>       mov    ah, 10 ; 6845 register for cursor set
9497 00003318 66890D[B36E0000]  <1>       mov    [CURSOR_MODE], cx ; save in data area
9498 0000331F E821F0FFFF        <1>       call   m16    ; output cx register
9499                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, lines);
9500 00003324 FEC7              <1>       inc    bh ; lines
9501 00003326 883D[9E6E0000]    <1>       mov    [CHAR_HEIGHT], bh
9502                            <1>       ;  outb(crtc_addr, 0x12);
9503 0000332C 66BAD403          <1>       mov    dx, 3D4h ; CRTC_ADDRESS
9504 00003330 B012              <1>       mov    al, 12h
9505 00003332 EE                <1>       out    dx, al
9506                            <1>       ; vde = inb(crtc_addr+1);
9507 00003333 6642              <1>       inc    dx
9508 00003335 EC                <1>       in     al, dx
```

```
9509 00003336 88C4              <1>       mov    ah, al
9510                            <1>       ; outb(crtc_addr, 0x07);
9511 00003338 664A              <1>       dec    dx
9512 0000333A B007              <1>       mov    al, 07h
9513 0000333C EE                <1>       out    dx, al
9514                            <1>       ; ovl = inb(crtc_addr+1);
9515 0000333D 6642              <1>       inc    dx
9516 0000333F EC                <1>       in     al, dx
9517                            <1>       ; vde += (((ovl & 0x02) << 7) + ((ovl & 0x40) << 3) + 1);
9518 00003340 88E2              <1>       mov    dl, ah ; vde
9519 00003342 88C6              <1>       mov    dh, al ; ovl
9520 00003344 6683E002          <1>       and    ax, 02h
9521 00003348 66C1E007          <1>       shl    ax, 7
9522 0000334C 6689C1            <1>       mov    cx, ax ; (ovl & 0x02) << 7)
9523 0000334F 88F0              <1>       mov    al, dh ; ovl
9524 00003351 6683E040          <1>       and    ax, 40h
9525 00003355 66C1E003          <1>       shl    ax, 3  ; (ovl & 0x40) << 3)
9526 00003359 6640              <1>       inc    ax ; + 1
9527 0000335B 6601C8            <1>       add    ax, cx
9528 0000335E 30F6              <1>       xor    dh, dh
9529 00003360 6601D0            <1>       add    ax, dx ; + vde
9530                            <1>       ; rows = vde / lines;
9531 00003363 F6F7              <1>       div    bh
9532                            <1>       ;dec   al ; rows -1
9533                            <1>       ; write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, rows-1);
9534 00003365 A2[A26E0000]      <1>       mov    [VGA_ROWS], al ; rows (not 'rows-1' !)
9535                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, rows * cols * 2);
9536                            <1>       ;mov   ah, [CRT_COLS]
9537                            <1>       ;mul   ah
9538                            <1>       ; 17/11/2020
9539 0000336A F625[9C6E0000]    <1>       mul    byte [CRT_COLS]
9540 00003370 66D1E0            <1>       shl    ax, 1
9541 00003373 66A3[88950100]    <1>       mov    [CRT_LEN], ax
9542 00003379 C3                <1>       retn
9543                            <1>
9544                            <1> load_text_8_14_pat:
9545                            <1>       ; 26/07/2016
9546                            <1>       ; 25/07/2016
9547                            <1>       ; 23/07/2016
9548                            <1>       ; 09/07/2016
9549                            <1>       ; load user defined (EGA/VGA) text fonts
9550                            <1>       ;
9551                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9552                            <1>       ; vgabios-0.7a (2011)
9553                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9554                            <1>       ; 'vgabios.c', 'biosfn_load_text_8_14_pat'
9555                            <1>
9556                            <1>       ; biosfn_load_text_8_14_pat (AL,BL)
9557                            <1>
9558                            <1>       ; get_font_access();
9559                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9560                            <1>       ; for(i=0;i<0x100;i++)
9561                            <1>       ; {
9562                            <1>       ;  src = i * 14;
9563                            <1>       ;  dest = blockaddr + i * 32;
9564                            <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont14+src, 14);
9565                            <1>       ; }
9566                            <1>       ; release_font_access();
9567                            <1>       ; if(AL>=0x10)
9568                            <1>       ; {
9569                            <1>       ; set_scan_lines(14);
9570                            <1>       ; }
9571                            <1>
9572 0000337A 50                <1>       push   eax
9573 0000337B E800FFFFFF        <1>       call   get_font_access
9574                            <1>
9575                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9576                            <1>       ;mov   dl, bl
9577                            <1>       ;and   dl, 3
9578                            <1>       ;shl   dx, 14
9579                            <1>       ;xchg  dx, bx
9580                            <1>       ;and   dl, 4
9581                            <1>       ;shl   dx, 11
9582                            <1>       ;add   dx, bx
9583                            <1>
9584                            <1>       ;xor   dx, dx  ; blockaddr = 0
9585                            <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
9586                            <1>
9587 00003380 28DB              <1>       sub    bl, bl ; i = 0
9588 00003382 B70E              <1>       mov    bh, 14
9589 00003384 BE[84670100]      <1>       mov    esi, vgafont14
9590 00003389 BF00000A00        <1>       mov    edi, 0A0000h
9591                            <1> lt8_14_1:
9592                            <1>       ;mov   al, bl
9593                            <1>       ;mul   bh
9594                            <1>       ;movzx esi, ax
9595                            <1>       ;add   esi, vgafont14
9596                            <1>       ;mov   al, bl
9597                            <1>       ;sub   ah, ah
9598                            <1>       ;shl   ax, 5 ; * 32
9599                            <1>       ;;add  ax, dx ; blockaddr + i * 32;
9600                            <1>       ;movzx edi, ax ; dest
9601                            <1>       ;add   edi, 0A0000h
9602 0000338E 0FB6CF            <1>       movzx  ecx, bh
9603 00003391 F3A4              <1>       rep    movsb
9604 00003393 83C712            <1>       add    edi, 18 ; 32 - 14
9605 00003396 FEC3              <1>       inc    bl
9606 00003398 75F4              <1>       jnz    short lt8_14_1
9607                            <1>       ;
9608 0000339A E816FFFFFF        <1>       call   release_font_access
9609                            <1>       ;
9610 0000339F 58                <1>       pop    eax
9611                            <1>       ; if(AL>=0x10)
9612 000033A0 3C10              <1>       cmp    al, 10h
9613 000033A2 7205              <1>       jb     short lt8_14_4
```

```
9614                                    <1>       ; BH = 14
9615                                    <1>       ; set_scan_lines(14);
9616 000033A4 E84BFFFFFF                <1>       call   set_scan_lines
9617                                    <1> lt8_14_4:
9618 000033A9 C3                        <1>       retn
9619                                    <1>
9620                                    <1> load_text_8_8_pat:
9621                                    <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
9622                                    <1>       ; 26/07/2016
9623                                    <1>       ; 25/07/2016
9624                                    <1>       ; 23/07/2016
9625                                    <1>       ; 09/07/2016
9626                                    <1>       ; load user defined (EGA/VGA) text fonts
9627                                    <1>       ;
9628                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
9629                                    <1>       ; vgabios-0.7a (2011)
9630                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
9631                                    <1>       ; 'vgabios.c', 'biosfn_load_text_8_8_pat'
9632                                    <1>
9633                                    <1>       ; biosfn_load_text_8_8_pat (AL,BL)
9634                                    <1>
9635                                    <1>       ; get_font_access();
9636                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9637                                    <1>       ; for(i=0;i<0x100;i++)
9638                                    <1>       ; {
9639                                    <1>       ;  src = i * 8;
9640                                    <1>       ;  dest = blockaddr + i * 32;
9641                                    <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont8+src, 8);
9642                                    <1>       ; }
9643                                    <1>       ; release_font_access();
9644                                    <1>       ; if(AL>=0x10)
9645                                    <1>       ; {
9646                                    <1>       ; set_scan_lines(8);
9647                                    <1>       ; }
9648                                    <1>
9649 000033AA 50                        <1>       push   eax
9650 000033AB E8D0FEFFFF                <1>       call   get_font_access
9651                                    <1>
9652                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9653                                    <1>       ;mov   dl, bl
9654                                    <1>       ;and   dl, 3
9655                                    <1>       ;shl   dx, 14
9656                                    <1>       ;xchg  dx, bx
9657                                    <1>       ;and   dl, 4
9658                                    <1>       ;shl   dx, 11
9659                                    <1>       ;add   dx, bx
9660                                    <1>
9661                                    <1>       ;xor   dx, dx  ; blockaddr = 0
9662                                    <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
9663                                    <1>
9664 000033B0 28DB                     <1>       sub    bl, bl ; i = 0
9665 000033B2 B708                     <1>       mov    bh, 8
9666                                    <1>       ;mov   esi, vgafont8
9667 000033B4 BF00000A00               <1>       mov    edi, 0A0000h
9668                                    <1>
9669                                    <1>       ; 05/01/2021
9670 000033B9 F605[7E120300]80         <1>       test   byte [ufont], 80h
9671 000033C0 7410                     <1>       jz     short lt8_8_0
9672                                    <1>              ; user font permission (after set mode)
9673 000033C2 F605[7E120300]08         <1>       test   byte [ufont], 8
9674 000033C9 7407                     <1>       jz     short lt8_8_0
9675 000033CB BE00500900               <1>       mov    esi, VGAFONT8USER
9676 000033D0 EB05                     <1>       jmp    short lt8_8_1
9677                                    <1> lt8_8_0:
9678 000033D2 BE[845F0100]             <1>       mov    esi, vgafont8
9679                                    <1> lt8_8_1:
9680                                    <1>       ;mov   al, bl
9681                                    <1>       ;mul   bh
9682                                    <1>       ;movzx esi, ax
9683                                    <1>       ;add   esi, vgafont8
9684                                    <1>       ;mov   al, bl
9685                                    <1>       ;sub   ah, ah
9686                                    <1>       ;shl   ax, 5 ; * 32
9687                                    <1>       ;;add  ax, dx ; blockaddr + i * 32;
9688                                    <1>       ;movzx edi, ax ; dest
9689                                    <1>       ;add   edi, 0A0000h
9690 000033D7 0FB6CF                  <1>       movzx  ecx, bh
9691 000033DA F3A4                     <1>       rep    movsb
9692 000033DC 83C718                   <1>       add    edi, 24 ; 32 - 8
9693 000033DF FEC3                     <1>       inc    bl
9694 000033E1 75F4                     <1>       jnz    short lt8_8_1
9695                                    <1>       ;
9696 000033E3 E8CDFEFFFF               <1>       call   release_font_access
9697                                    <1>       ;
9698 000033E8 58                       <1>       pop    eax
9699                                    <1>       ; if(AL>=0x10)
9700 000033E9 3C10                     <1>       cmp    al, 10h
9701 000033EB 7205                     <1>       jb     short lt8_8_2
9702                                    <1>       ; BH = 8
9703                                    <1>       ; set_scan_lines(8);
9704 000033ED E802FFFFFF               <1>       call   set_scan_lines
9705                                    <1> lt8_8_2:
9706 000033F2 C3                       <1>       retn
9707                                    <1>
9708                                    <1> load_text_8_16_pat:
9709                                    <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
9710                                    <1>       ; 26/07/2016
9711                                    <1>       ; 25/07/2016
9712                                    <1>       ; 23/07/2016
9713                                    <1>       ; 09/07/2016
9714                                    <1>       ; load user defined (EGA/VGA) text fonts
9715                                    <1>       ;
9716                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
9717                                    <1>       ; vgabios-0.7a (2011)
9718                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
```

```
9719                                    <1>       ; 'vgabios.c', 'biosfn_load_text_8_16_pat'
9720                                    <1>
9721                                    <1>       ; biosfn_load_text_8_16_pat (AL,BL)
9722                                    <1>
9723                                    <1>       ; get_font_access();
9724                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9725                                    <1>       ; for(i=0;i<0x100;i++)
9726                                    <1>       ; {
9727                                    <1>       ;  src = i * 16;
9728                                    <1>       ;  dest = blockaddr + i * 32;
9729                                    <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont16+src, 16);
9730                                    <1>       ; }
9731                                    <1>       ; release_font_access();
9732                                    <1>       ; if(AL>=0x10)
9733                                    <1>       ; {
9734                                    <1>       ; set_scan_lines(16);
9735                                    <1>       ; }
9736                                    <1>
9737 000033F3 50                       <1>       push   eax
9738 000033F4 E887FEFFFF               <1>       call   get_font_access
9739                                    <1>
9740                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9741                                    <1>       ;mov   dl, bl
9742                                    <1>       ;and   dl, 3
9743                                    <1>       ;shl   dx, 14
9744                                    <1>       ;xchg  dx, bx
9745                                    <1>       ;and   dl, 4
9746                                    <1>       ;shl   dx, 11
9747                                    <1>       ;add   dx, bx
9748                                    <1>
9749                                    <1>       ;xor   dx, dx  ; blockaddr = 0
9750                                    <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
9751                                    <1>
9752 000033F9 28DB                     <1>       sub    bl, bl ; i = 0
9753 000033FB B710                     <1>       mov    bh, 16
9754                                    <1>       ;mov   esi, vgafont16
9755 000033FD BF00000A00               <1>       mov    edi, 0A0000h
9756 00003402 0FB6C7                   <1>       movzx  eax, bh
9757                                    <1>
9758                                    <1>       ; 05/01/2021
9759 00003405 F605[7E120300]80         <1>       test   byte [ufont], 80h
9760 0000340C 7410                     <1>       jz     short lt8_16_0
9761                                    <1>              ; user font permission (after set mode)
9762 0000340E F605[7E120300]10         <1>       test   byte [ufont], 16
9763 00003415 7407                     <1>       jz     short lt8_16_0
9764 00003417 BE00400900               <1>       mov    esi, VGAFONT16USER
9765 0000341C EB05                     <1>       jmp    short lt8_16_1
9766                                    <1> lt8_16_0:
9767 0000341E BE[84750100]             <1>       mov    esi, vgafont16
9768                                    <1> lt8_16_1:
9769                                    <1>       ;mov   al, bl
9770                                    <1>       ;mul   bh
9771                                    <1>       ;movzx esi, ax
9772                                    <1>       ;add   esi, vgafont16
9773                                    <1>       ;mov   al, bl ; i
9774                                    <1>       ;sub   ah, ah
9775                                    <1>       ;shl   ax, 5 ; * 32
9776                                    <1>       ;;add  ax, dx ; blockaddr + i * 32;
9777                                    <1>       ;movzx edi, ax ; dest
9778                                    <1>       ;add   edi, 0A0000h
9779                                    <1>       ;movzx ecx, bh
9780 00003423 89C1                     <1>       mov    ecx, eax ; 16
9781 00003425 F3A4                     <1>       rep    movsb
9782 00003427 01C7                     <1>       add    edi, eax ; add edi, 16
9783 00003429 FEC3                     <1>       inc    bl
9784 0000342B 75F6                     <1>       jnz    short lt8_16_1
9785                                    <1>       ;
9786 0000342D E883FEFFFF               <1>       call   release_font_access
9787                                    <1>       ;
9788 00003432 58                       <1>       pop    eax
9789                                    <1>       ; if(AL>=0x10)
9790 00003433 3C10                     <1>       cmp    al, 10h
9791 00003435 7205                     <1>       jb     short lt8_16_2
9792                                    <1>       ; BH = 16
9793                                    <1>       ; set_scan_lines(16);
9794 00003437 E8B8FEFFFF               <1>       call   set_scan_lines
9795                                    <1> lt8_16_2:
9796 0000343C C3                       <1>       retn
9797                                    <1>
9798                                    <1> load_gfx_user_chars:
9799                                    <1>       ; 08/01/2021
9800                                    <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
9801                                    <1>       ; 08/08/2016
9802                                    <1>       ; 10/07/2016
9803                                    <1>       ; Setup User-Defined Font for Graphics Mode (VGA)
9804                                    <1>       ;
9805                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
9806                                    <1>       ; vgabios-0.7a (2011)
9807                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
9808                                    <1>       ; 'vgabios.c', 'biosfn_load_gfx_user_chars'
9809                                    <1>
9810                                    <1>       ; biosfn_load_gfx_user_chars (ES,BP,CX,BL,DL)
9811                                    <1>       ; /* set 0x43 INT pointer */
9812                                    <1>       ; write_word(0x0, 0x43*4, BP);
9813                                    <1>       ; write_word(0x0, 0x43*4+2, ES);
9814                                    <1>
9815                                    <1>       ; 08/01/2021
9816                                    <1>
9817                                    <1>       ; BL   screen rows code: 00H = user-specified (in DL)
9818                                    <1>         ;                      01H = 14 rows
9819                                    <1>         ;                      02H = 25 rows
9820                                    <1>         ;                      03H = 43 rows
9821                                    <1>         ; CX   bytes per character definition
9822                                    <1>         ; DL   (when BL=0) custom number of character rows on screen
9823                                    <1>         ; EBP  address of font-definition information (user's mem space)
```

```
                                <1>
                                <1>       ; 05/01/2021
                                <1>       ;xor   eax, eax
                                <1>       ;dec   eax ; 0FFFFFFFFh (user defined fonts)
                                <1>       ;mov   [VGA_INT43H], eax
                                <1>
                                <1>       ; 08/01/2021
                                <1>       ; ebp = video font data (buffer) address
0000343D 892D[9A950100]         <1>       mov    [VGA_INT43H], ebp
                                <1>
                                <1>       ; switch (BL) {
                                <1>       ; case 0:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                                <1>       ;    break;
00003443 20DB                   <1>       and    bl, bl
00003445 7508                   <1>       jnz    short l_gfx_uc_1
00003447 8815[A26E0000]         <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
0000344D EB23                   <1>       jmp    short l_gfx_uc_4
                                <1> l_gfx_uc_1:
                                <1>       ; case 1:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                                <1>       ;    break;
0000344F FECB                   <1>       dec    bl
00003451 7509                   <1>       jnz    short l_gfx_uc_2
                                <1>       ; bl = 1
00003453 C605[A26E0000]0E       <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
0000345A EB16                   <1>       jmp    short l_gfx_uc_4
                                <1> l_gfx_uc_2:
0000345C FECB                   <1>       dec    bl
0000345E 740B                   <1>       jz     short l_gfx_uc_3 ; bl = 2
00003460 FECB                   <1>       dec    bl
00003462 750E                   <1>       jnz    short l_gfx_uc_4 ; bl > 3
                                <1>       ; bl = 3
                                <1>       ; case 3:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                                <1>       ;    break;
00003464 C605[A26E0000]2B       <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
                                <1> l_gfx_uc_3:
                                <1>       ; case 2:
                                <1>       ; default:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
                                <1>       ;    break;
                                <1>       ; bl = 2 or bl > 3
0000346B C605[A26E0000]19       <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
                                <1>       ; }
                                <1> l_gfx_uc_4:
                                <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, CX);
00003472 880D[9E6E0000]         <1>       mov    [CHAR_HEIGHT], cl
                                <1>       ; }
00003478 C3                     <1>       retn
                                <1>
                                <1> load_gfx_8_14_chars:
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                                <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_gfx_8_14_chars'
                                <1>
                                <1>       ; biosfn_load_gfx_8_14_chars (BL)
                                <1>       ; /* set 0x43 INT pointer */
                                <1>       ; write_word(0x0, 0x43*4, &vgafont14);
                                <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
00003479 C705[9A950100]-        <1>       mov    dword [VGA_INT43H], vgafont14
0000347F [84670100]             <1>
                                <1>
                                <1>       ; BL    screen rows code: 00H = user-specified (in DL)
                                <1>         ;                       01H = 14 rows
                                <1>         ;                       02H = 25 rows
                                <1>         ;                       03H = 43 rows
                                <1>         ; DL    (when BL=0) custom number of char rows on screen
                                <1>
                                <1>       ; switch (BL) {
                                <1>       ; case 0:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                                <1>       ;    break;
00003483 20DB                   <1>       and    bl, bl
00003485 7508                   <1>       jnz    short l_gfx_8_14c_1
00003487 8815[A26E0000]         <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
0000348D EB23                   <1>       jmp    short l_gfx_8_14c_4
                                <1> l_gfx_8_14c_1:
                                <1>       ; case 1:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                                <1>       ;    break;
0000348F FECB                   <1>       dec    bl
00003491 7509                   <1>       jnz    short l_gfx_8_14c_2
                                <1>       ; bl = 1
00003493 C605[A26E0000]0E       <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
0000349A EB16                   <1>       jmp    short l_gfx_8_14c_4
                                <1> l_gfx_8_14c_2:
0000349C FECB                   <1>       dec    bl
0000349E 740B                   <1>       jz     short l_gfx_8_14c_3 ; bl = 2
000034A0 FECB                   <1>       dec    bl
000034A2 750E                   <1>       jnz    short l_gfx_8_14c_4 ; bl > 3
                                <1>       ; bl = 3
                                <1>       ; case 3:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                                <1>       ;    break;
000034A4 C605[A26E0000]2B       <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
                                <1> l_gfx_8_14c_3:
                                <1>       ; case 2:
                                <1>       ; default:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
```

```
                                <1>       ;    break;
                                <1>       ; bl = 2 or bl > 3
000034AB C605[A26E0000]19       <1>       mov   byte [VGA_ROWS], 25  ; not 24 !
                                <1>       ; }
                                <1> l_gfx_8_14c_4:
                                <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 14);
000034B2 C605[9E6E0000]0E       <1>         mov     byte [CHAR_HEIGHT], 14
                                <1>       ; }
000034B9 C3                     <1>       retn
                                <1>
                                <1> load_gfx_8_8_chars:
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                                <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_gfx_8_8_dd_chars'
                                <1>
                                <1>       ; biosfn_load_gfx_8_8_dd_chars (BL)
                                <1>       ; /* set 0x43 INT pointer */
                                <1>       ; write_word(0x0, 0x43*4, &vgafont8);
                                <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
000034BA C705[9A950100]-        <1>       mov   dword [VGA_INT43H], vgafont8
000034C0 [845F0100]             <1>
                                <1>
                                <1>       ; BL    screen rows code: 00H = user-specified (in DL)
                                <1>         ;                         01H = 14 rows
                                <1>         ;                         02H = 25 rows
                                <1>         ;                         03H = 43 rows
                                <1>         ; DL    (when BL=0) custom number of char rows on screen
                                <1>
                                <1>       ; switch (BL) {
                                <1>       ; case 0:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                                <1>       ;    break;
000034C4 20DB                   <1>       and   bl, bl
000034C6 7508                   <1>       jnz   short l_gfx_8_8c_1
000034C8 8815[A26E0000]         <1>       mov   [VGA_ROWS], dl  ; not DL-1 !
000034CE EB23                   <1>       jmp   short l_gfx_8_8c_4
                                <1> l_gfx_8_8c_1:
                                <1>       ; case 1:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                                <1>       ;    break;
000034D0 FECB                   <1>       dec   bl
000034D2 7509                   <1>       jnz   short l_gfx_8_8c_2
                                <1>       ; bl = 1
000034D4 C605[A26E0000]0E       <1>       mov   byte [VGA_ROWS], 14  ; not 13 !
000034DB EB16                   <1>       jmp   short l_gfx_8_8c_4
                                <1> l_gfx_8_8c_2:
000034DD FECB                   <1>       dec   bl
000034DF 740B                   <1>       jz    short l_gfx_8_8c_3 ; bl = 2
000034E1 FECB                   <1>       dec   bl
000034E3 750E                   <1>       jnz   short l_gfx_8_8c_4 ; bl > 3
                                <1>       ; bl = 3
                                <1>       ; case 3:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                                <1>       ;    break;
000034E5 C605[A26E0000]2B       <1>       mov   byte [VGA_ROWS], 43  ; not 42 !
                                <1> l_gfx_8_8c_3:
                                <1>       ; case 2:
                                <1>       ; default:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
                                <1>       ;    break;
                                <1>       ; bl = 2 or bl > 3
000034EC C605[A26E0000]19       <1>       mov   byte [VGA_ROWS], 25  ; not 24 !
                                <1>       ; }
                                <1> l_gfx_8_8c_4:
                                <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 8);
000034F3 C605[9E6E0000]08       <1>         mov     byte [CHAR_HEIGHT], 8
                                <1>       ; }
000034FA C3                     <1>       retn
                                <1>
                                <1> load_gfx_8_16_chars:
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                                <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_gfx_8_16_chars'
                                <1>
                                <1>       ; biosfn_load_gfx_8_16_chars (BL)
                                <1>       ; /* set 0x43 INT pointer */
                                <1>       ; write_word(0x0, 0x43*4, &vgafont16);
                                <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
000034FB C705[9A950100]-        <1>       mov   dword [VGA_INT43H], vgafont16
00003501 [84750100]             <1>
                                <1>
                                <1>       ; BL    screen rows code: 00H = user-specified (in DL)
                                <1>         ;                         01H = 14 rows
                                <1>         ;                         02H = 25 rows
                                <1>         ;                         03H = 43 rows
                                <1>         ; DL    (when BL=0) custom number of char rows on screen
                                <1>
                                <1>       ; switch (BL) {
                                <1>       ; case 0:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                                <1>       ;    break;
00003505 20DB                   <1>       and   bl, bl
00003507 7508                   <1>       jnz   short l_gfx_8_16c_1
00003509 8815[A26E0000]         <1>       mov   [VGA_ROWS], dl  ; not DL-1 !
0000350F EB23                   <1>       jmp   short l_gfx_8_16c_4
```

```
10031                                   <1> l_gfx_8_16c_1:
10032                                   <1>         ; case 1:
10033                                   <1>         ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
10034                                   <1>         ;    break;
10035 00003511 FECB                       <1>         dec    bl
10036 00003513 7509                       <1>         jnz    short l_gfx_8_16c_2
10037                                   <1>         ; bl = 1
10038 00003515 C605[A26E0000]0E           <1>         mov    byte [VGA_ROWS], 14  ; not 13 !
10039 0000351C EB16                       <1>         jmp    short l_gfx_8_16c_4
10040                                   <1> l_gfx_8_16c_2:
10041 0000351E FECB                       <1>         dec    bl
10042 00003520 740B                       <1>         jz     short l_gfx_8_16c_3 ; bl = 2
10043 00003522 FECB                       <1>         dec    bl
10044 00003524 750E                       <1>         jnz    short l_gfx_8_16c_4 ; bl > 3
10045                                   <1>         ; bl = 3
10046                                   <1>         ; case 3:
10047                                   <1>         ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
10048                                   <1>         ;    break;
10049 00003526 C605[A26E0000]2B           <1>         mov    byte [VGA_ROWS], 43  ; not 42 !
10050                                   <1> l_gfx_8_16c_3:
10051                                   <1>         ; case 2:
10052                                   <1>         ; default:
10053                                   <1>         ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
10054                                   <1>         ;    break;
10055                                   <1>         ; bl = 2 or bl > 3
10056 0000352D C605[A26E0000]19           <1>         mov    byte [VGA_ROWS], 25  ; not 24 !
10057                                   <1>         ; }
10058                                   <1> l_gfx_8_16c_4:
10059                                   <1>         ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 16);
10060 00003534 C605[9E6E0000]10           <1>           mov     byte [CHAR_HEIGHT], 16
10061                                   <1>         ; }
10062 0000353B C3                         <1>         retn
10063                                   <1>
10064                                   <1> get_font_info:
10065                                   <1>         ; 08/01/2021 (TRDOS 386 v2.0.3)
10066                                   <1>         ; 19/09/2016
10067                                   <1>         ; 08/08/2016
10068                                   <1>         ; 10/07/2016
10069                                   <1>         ; Get Current Character Generator Info (VGA)
10070                                   <1>         ;
10071                                   <1>         ; derived from 'Plex86/Bochs VGABios' source code
10072                                   <1>         ; vgabios-0.7a (2011)
10073                                   <1>         ; by the LGPL VGABios developers Team (2001-2008)
10074                                   <1>         ; 'vgabios.c', 'biosfn_get_font_info'
10075                                   <1>
10076                                   <1>         ; Modified for TRDOS 386 !
10077                                   <1>         ;
10078                                   <1>         ; INPUT ->
10079                                   <1>         ;    AX = 1130h
10080                                   <1>         ;    BL = 0 -> Get info for current VGA font
10081                                   <1>         ;              (BH = unused)
10082                                   <1>         ;    19/09/2016
10083                                   <1>         ;    BL > 0 -> Get requested character font data
10084                                   <1>         ;       BL = 1 -> vgafont8
10085                                   <1>         ;        BL = 2 -> vgafont14
10086                                   <1>         ;       BL = 3 -> vgafont16
10087                                   <1>         ;      ;08/01/2021
10088                                   <1>         ;       BL = 4 -> user defined 8x8 font
10089                                   <1>         ;       BL = 5 -> user defined 8x14 font
10090                                   <1>         ;       BL = 6 -> user defined 8x16 font
10091                                   <1>         ;       BL > 6 -> Invalid function (for now!)
10092                                   <1>         ;       BH = ASCII code of the first character
10093                                   <1>         ;       ECX = Number of characters from the 1st char
10094                                   <1>         ;       ECX >= 256 -> All (256-BH) characters
10095                                   <1>         ;       ECX = 0 -> All characters (BH = unused)
10096                                   <1>         ;        EDX = User's Buffer Address
10097                                   <1>         ; OUTPUT ->
10098                                   <1>         ;    AL = height (scanlines), bytes per character
10099                                   <1>         ;    AH = screen rows
10100                                   <1>         ;    Byte 16-23 of EAX = number of columns
10101                                   <1>         ;    Byte 24-31 of EAX =
10102                                   <1>         ;        0 -> default font (not configured yet)
10103                                   <1>         ;        0FFh -> user defined font
10104                                   <1>         ;        14 = vgafont14
10105                                   <1>         ;         8 = vgafont8
10106                                   <1>         ;        16 = vgafont16
10107                                   <1>         ;   If BL input > 0 ->
10108                                   <1>         ;       EAX = Actual transfer count
10109                                   <1>         ;
10110 0000353C 20DB                       <1>         and    bl, bl
10111 0000353E 740F                       <1>         jz     short gfi_1
10112                                   <1>         ; invalid function (input)
10113                                   <1>         ; 08/01/2021
10114 00003540 80FB04                     <1>         cmp    bl, 4
10115 00003543 7263                       <1>         jb     short gfi_5
10116 00003545 7441                       <1>         je     short gfi_3
10117 00003547 80FB06                     <1>         cmp    bl, 6
10118 0000354A 744C                       <1>         je     short gfi_4
10119                                   <1>         ; bh = 5 or bh > 6
10120                                   <1> gfi_0:
10121 0000354C 31C0                       <1>         xor    eax, eax ; 0
10122 0000354E C3                         <1>         retn
10123                                   <1> gfi_1:
10124 0000354F A0[9E6E0000]               <1>         mov    al, [CHAR_HEIGHT]
10125 00003554 8A25[A26E0000]             <1>         mov    ah, [VGA_ROWS]
10126 0000355A C1E010                     <1>         shl    eax, 16
10127 0000355D A0[9C6E0000]               <1>         mov    al, [CRT_COLS]
10128 00003562 8B0D[9A950100]             <1>         mov    ecx, [VGA_INT43H]
10129 00003568 21C9                       <1>         and    ecx, ecx
10130 0000356A 7418                       <1>         jz     short gfi_2 ; 0 = default font
10131                                   <1>         ; 08/01/2021
10132 0000356C FECC                       <1>         dec    ah ; 0FFh
10133 0000356E 81F900400900               <1>         cmp    ecx, VGAFONT16USER
10134 00003574 740E                       <1>         je     short gfi_2
10135 00003576 81F900500900               <1>         cmp    ecx, VGAFONT8USER
```

```
10136 0000357C 7406               <1>       je    short gfi_2
10137 0000357E 8A25[9E6E0000]     <1>       mov   ah, [CHAR_HEIGHT] ; font size = height
10138                             <1> gfi_2:
10139 00003584 C1C010             <1>       rol   eax, 16
10140 00003587 C3                 <1>       retn
10141                             <1> gfi_3:
10142                             <1>       ; 08/01/2021
10143 00003588 F605[7E120300]08   <1>       test  byte [ufont], 08h ; 8x8 user font
10144 0000358F 74BB               <1>       jz    short gfi_0 ; not loaded !
10145 00003591 BE00500900         <1>       mov   esi, VGAFONT8USER ; *
10146                             <1>       ;mov  bl, 8
10147                             <1>       ;jmp  short gfi_8
10148 00003596 EB4D               <1>       jmp   short gfi_10
10149                             <1> gfi_4:
10150                             <1>       ; 08/01/2021
10151 00003598 F605[7E120300]10   <1>       test  byte [ufont], 10h ; 8x16 user font
10152 0000359F 74AB               <1>       jz    short gfi_0 ; not loaded !
10153 000035A1 BE00400900         <1>       mov   esi, VGAFONT16USER ; *
10154 000035A6 EB15               <1>       jmp   short gfi_7
10155                             <1> gfi_5:
10156 000035A8 80FB02             <1>       cmp   bl, 2
10157 000035AB 7233               <1>       jb    short gfi_9
10158 000035AD 7709               <1>       ja    short gfi_6
10159                             <1>       ;BL = 2 -> vgafont14
10160 000035AF BE[84670100]       <1>       mov   esi, vgafont14 ; *
10161 000035B4 B30E               <1>       mov   bl, 14
10162 000035B6 EB07               <1>       jmp   short gfi_8
10163                             <1> gfi_6:
10164                             <1>       ;BL = 3 -> vgafont16
10165 000035B8 BE[84750100]       <1>       mov   esi, vgafont16 ; *
10166                             <1> gfi_7:
10167 000035BD B310               <1>       mov   bl, 16
10168                             <1> gfi_8:
10169 000035BF 89D7               <1>       mov   edi, edx ; **
10170 000035C1 09C9               <1>       or    ecx, ecx
10171 000035C3 7424               <1>       jz    short gfi_11 ; all chars from the 00h
10172 000035C5 88F8               <1>       mov   al, bh ; character index
10173 000035C7 F6E3               <1>       mul   bl ; char index * char height/size
10174 000035C9 0FB7D0             <1>       movzx edx, ax
10175 000035CC 01D6               <1>       add   esi, edx ; *
10176 000035CE 66BAFF00           <1>       mov   dx, 255
10177 000035D2 28FA               <1>       sub   dl, bh
10178 000035D4 6642               <1>       inc   dx
10179 000035D6 39D1               <1>       cmp   ecx, edx
10180 000035D8 770F               <1>       ja    short gfi_11
10181 000035DA 7412               <1>       je    short gfi_12
10182 000035DC 89D1               <1>       mov   ecx, edx
10183 000035DE EB0E               <1>       jmp   short gfi_12
10184                             <1> gfi_9:
10185                             <1>       ;BL = 1 -> vgafont8
10186 000035E0 BE[845F0100]       <1>       mov   esi, vgafont8 ; *
10187                             <1> gfi_10:
10188 000035E5 B308               <1>       mov   bl, 8
10189 000035E7 EBD6               <1>       jmp   short gfi_8
10190                             <1> gfi_11:
10191 000035E9 B900010000         <1>       mov   ecx, 256
10192                             <1> gfi_12:
10193                             <1>       ; 08/01/2021
10194 000035EE 89C8               <1>       mov   eax, ecx ; character count
10195 000035F0 30FF               <1>       xor   bh, bh
10196 000035F2 66F7E3             <1>       mul   bx ; char count * char height/size
10197 000035F5 89C1               <1>       mov   ecx, eax
10198                             <1>
10199                             <1>       ; ESI = source address in system space
10200                             <1>       ; EDI = user's buffer address
10201                             <1>       ; ECX = transfer (byte) count
10202 000035F7 E828E40000         <1>       call  transfer_to_user_buffer
10203 000035FC 89C8               <1>       mov   eax, ecx ; actual transfer count
10204 000035FE C3                 <1>       retn
10205                             <1>
10206                             <1> vga_pal_funcs:
10207                             <1>       ; 10/08/2016
10208                             <1>       ; VGA Palette functions
10209                             <1>       ;
10210                             <1>       ; derived from 'Plex86/Bochs VGABios' source code
10211                             <1>       ; vgabios-0.7a (2011)
10212                             <1>       ; by the LGPL VGABios developers Team (2001-2008)
10213                             <1>       ; 'vgabios.c', 'vgarom.asm'
10214                             <1>
10215 000035FF 3C00              <1>       cmp   al, 0
10216 00003601 0F848F000000       <1>         je      set_single_palette_reg
10217                             <1> vga_palf_1001:
10218 00003607 3C01               <1>       cmp   al, 1
10219 00003609 0F84B0000000       <1>         je      set_overscan_border_color
10220                             <1> vga_palf_1002:
10221 0000360F 3C02               <1>       cmp   al, 2
10222 00003611 0F84AC000000       <1>         je      set_all_palette_reg
10223                             <1> vga_palf_1003:
10224 00003617 3C03               <1>       cmp   al, 3
10225 00003619 0F84E4000000       <1>         je      toggle_intensity
10226                             <1> vga_palf_1007:
10227 0000361F 3C07               <1>       cmp   al, 7
10228 00003621 0F8409010000       <1>         je      get_single_palette_reg
10229 00003627 7266               <1>       jb    short vga_palf_unknown
10230                             <1> vga_palf_1008:
10231 00003629 3C08               <1>       cmp   al, 8
10232 0000362B 0F8433010000       <1>         je      read_overscan_border_color
10233                             <1> vga_palf_1009:
10234 00003631 3C09               <1>       cmp   al, 9
10235 00003633 0F842F010000       <1>       je    get_all_palette_reg
10236                             <1> vga_palf_1010:
10237 00003639 3C10               <1>       cmp   al, 10h
10238 0000363B 0F8483010000       <1>       je    set_single_dac_reg
10239 00003641 724C               <1>       jb    short vga_palf_unknown
10240                             <1> vga_palf_1012:
```

```
10241 00003643 3C12              <1>       cmp    al, 12h
10242 00003645 0F8492010000      <1>         je      set_all_dac_reg
10243 0000364B 7242              <1>       jb     short vga_palf_unknown
10244                            <1> vga_palf_1013:
10245 0000364D 3C13              <1>       cmp    al, 13h
10246 0000364F 0F84C6010000      <1>         je      select_video_dac_color_page
10247                            <1> vga_palf_1015:
10248 00003655 3C15              <1>       cmp    al, 15h
10249 00003657 0F840A020000      <1>       je     read_single_dac_reg
10250 0000365D 7230              <1>       jb     short vga_palf_unknown
10251                            <1> vga_palf_1017:
10252 0000365F 3C17              <1>       cmp    al, 17h
10253 00003661 0F8420020000      <1>         je      read_all_dac_reg
10254 00003667 7226              <1>       jb     short vga_palf_unknown
10255                            <1> vga_palf_1018:
10256 00003669 3C18              <1>       cmp    al, 18h
10257 0000366B 0F8456020000      <1>         je      set_pel_mask
10258                            <1> vga_palf_1019:
10259 00003671 3C19              <1>       cmp    al, 19h
10260 00003673 0F845A020000      <1>         je      read_pel_mask
10261                            <1> vga_palf_101A:
10262 00003679 3C1A              <1>       cmp    al, 1Ah
10263 0000367B 0F8460020000      <1>         je      read_video_dac_state
10264                            <1> vga_palf_101B:
10265 00003681 3C1B              <1>       cmp    al, 1Bh
10266                            <1>       ;jne   short vga_palf_unknown
10267 00003683 770A              <1>       ja     short vga_palf_unknown
10268                            <1>
10269 00003685 E8CDF5FFFF        <1>       call   gray_scale_summing
10270 0000368A E962E4FFFF        <1>         jmp     VIDEO_RETURN
10271                            <1>
10272                            <1> vga_palf_unknown:
10273 0000368F 29C0              <1>       sub    eax, eax ; 0 = invalid function
10274 00003691 E960E4FFFF        <1>         jmp     _video_return
10275                            <1>
10276                            <1> set_single_palette_reg:
10277                            <1>       ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
10278                            <1>       ; 10/08/2016
10279                            <1>       ; Set One Palette Register
10280                            <1>       ; BL = register number to set
10281                            <1>       ;     (a 4-bit attribute nibble: 00h-0Fh)
10282                            <1>       ; BH = 6-bit RGB color to display
10283                            <1>       ;      for that attribute
10284                            <1>
10285 00003696 80FB14            <1>       cmp    bl, 14h
10286                            <1>       ;ja    short no_actl_reg1
10287 00003699 0F8752E4FFFF      <1>       ja     VIDEO_RETURN
10288                            <1>       ;push  ax
10289                            <1>       ;push  dx
10290                            <1>       ; 12/04/2021
10291 0000369F 50                <1>       push   eax
10292 000036A0 52                <1>       push   edx
10293 000036A1 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10294 000036A5 EC                <1>       in     al, dx
10295 000036A6 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10296 000036AA 88D8              <1>       mov    al, bl
10297 000036AC EE                <1>       out    dx, al
10298 000036AD 88F8              <1>       mov    al, bh
10299 000036AF EE                <1>       out    dx, al
10300 000036B0 B020              <1>       mov    al, 20h
10301 000036B2 EE                <1>       out    dx, al
10302                            <1>       ; ifdef VBOX
10303 000036B3 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10304 000036B7 EC                <1>       in     al, dx
10305                            <1>       ; endif ; VBOX
10306                            <1>       ;pop   dx
10307                            <1>       ;pop   ax
10308                            <1>       ; 12/04/2021
10309 000036B8 5A                <1>       pop    edx
10310 000036B9 58                <1>       pop    eax
10311                            <1> ;no_actl_reg1:
10312 000036BA E932E4FFFF        <1>       jmp    VIDEO_RETURN
10313                            <1>
10314                            <1> set_overscan_border_color:
10315                            <1>       ; 10/08/2016
10316                            <1>       ; Set Overscan/Border Color Register
10317                            <1>       ; BH = 6-bit RGB color to display
10318                            <1>       ;      for that attribute
10319                            <1>
10320 000036BF B311              <1>       mov    bl, 11h
10321 000036C1 EBD3              <1>       jmp    short set_single_palette_reg
10322                            <1>
10323                            <1> set_all_palette_reg:
10324                            <1>       ; 10/08/2016
10325                            <1>       ; Set All Palette Registers and Overscan
10326                            <1>       ; EDX = Address of 17 bytes;
10327                            <1>       ; an rgbRGB value for each of 16 palette
10328                            <1>       ; registers plus one for the border.
10329                            <1>
10330 000036C3 89D6              <1>       mov    esi, edx ; user buffer
10331 000036C5 B911000000        <1>       mov    ecx, 17
10332 000036CA 89E7              <1>       mov    edi, esp
10333 000036CC 83EC14            <1>       sub    esp, 20
10334 000036CF E89AE30000        <1>       call   transfer_from_user_buffer
10335                            <1>       ;jc    VIDEO_RETURN
10336                            <1>
10337 000036D4 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10338 000036D8 EC                <1>       in     al, dx
10339 000036D9 B100              <1>       mov    cl, 0
10340 000036DB 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10341                            <1> set_palette_loop:
10342 000036DF 88C8              <1>       mov    al, cl
10343 000036E1 EE                <1>       out    dx, al
10344 000036E2 8A07              <1>       mov    al, [edi]
10345 000036E4 EE                <1>       out    dx, al
```

```
10346 000036E5 47                <1>       inc    edi
10347 000036E6 FEC1              <1>       inc    cl
10348 000036E8 80F910            <1>       cmp    cl, 10h
10349 000036EB 75F2              <1>       jne    short set_palette_loop
10350 000036ED B011              <1>       mov    al, 11h
10351 000036EF EE                <1>       out    dx, al
10352 000036F0 8A07              <1>       mov    al, [edi]
10353 000036F2 EE                <1>       out    dx, al
10354 000036F3 B020              <1>       mov    al, 20h
10355 000036F5 EE                <1>       out    dx, al
10356                            <1>       ; ifdef VBOX
10357 000036F6 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10358 000036FA EC                <1>       in     al, dx
10359                            <1>       ; endif ; VBOX
10360 000036FB 83C414            <1>       add    esp, 20
10361 000036FE E9EEE3FFFF        <1>       jmp    VIDEO_RETURN
10362                            <1>
10363                            <1> toggle_intensity:
10364                            <1>       ; 10/08/2016
10365                            <1>       ; Select Foreground Blink or Bold Background
10366                            <1>       ; BL = 00h = enable bold backgrounds
10367                            <1>       ;          (16 background colors)
10368                            <1>         ;      01h = enable blinking foreground
10369                            <1>       ;          (8 background colors)
10370                            <1>
10371 00003703 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10372 00003707 EC                <1>       in     al, dx
10373 00003708 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10374 0000370C B010              <1>       mov    al, 10h
10375 0000370E EE                <1>       out    dx, al
10376 0000370F 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10377 00003713 EC                <1>       in     al, dx
10378 00003714 24F7              <1>       and    al, 0F7h
10379 00003716 80E301            <1>       and    bl, 01h
10380 00003719 C0E303            <1>       shl    bl, 3
10381 0000371C 08D8              <1>       or     al, bl
10382 0000371E 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10383 00003722 EE                <1>       out    dx, al
10384 00003723 B020              <1>       mov    al, 20h
10385 00003725 EE                <1>       out    dx, al
10386                            <1>       ; ifdef VBOX
10387 00003726 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10388 0000372A EC                <1>       in     al, dx
10389                            <1>       ; endif ; VBOX
10390 0000372B E9C1E3FFFF        <1>       jmp    VIDEO_RETURN
10391                            <1>
10392                            <1> get_single_palette_reg:
10393                            <1>       ; 10/08/2016
10394                            <1>       ; Read One Palette Register
10395                            <1>         ; INPUT:
10396                            <1>       ; BL = Palette register to read (00h-0Fh)
10397                            <1>       ; OUTPUT:
10398                            <1>       ; BH = Current rgbRGB value of specified register
10399                            <1>       ;      for that attribute
10400                            <1>
10401 00003730 80FB14            <1>       cmp    bl, 14h
10402                            <1>       ;ja    short no_actl_reg2
10403 00003733 0F87B8E3FFFF      <1>       ja     VIDEO_RETURN
10404                            <1>
10405 00003739 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10406 0000373D EC                <1>       in     al, dx
10407 0000373E 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10408 00003742 88D8              <1>       mov    al, bl
10409 00003744 EE                <1>       out    dx, al
10410 00003745 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10411 00003749 EC                <1>       in     al, dx
10412 0000374A 8844240D          <1>       mov    [esp+13], al ; bh
10413 0000374E 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10414 00003752 EC                <1>       in     al, dx
10415 00003753 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10416 00003757 B020              <1>       mov    al, 20h
10417 00003759 EE                <1>       out    dx, al
10418                            <1>       ; ifdef VBOX
10419 0000375A 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10420 0000375E EC                <1>       in     al, dx
10421                            <1>       ; endif ; VBOX
10422 0000375F E98DE3FFFF        <1>       jmp    VIDEO_RETURN
10423                            <1>
10424                            <1> read_overscan_border_color:
10425                            <1>       ; 10/08/2016
10426                            <1>       ; Read Overscan Register
10427                            <1>       ; OUTPUT:
10428                            <1>       ; BH = current rgbRGB value
10429                            <1>       ;      of the overscan/border register
10430                            <1>
10431 00003764 B311              <1>       mov    bl, 11h
10432 00003766 EBC8              <1>       jmp    short get_single_palette_reg
10433                            <1>
10434                            <1> get_all_palette_reg:
10435                            <1>       ; 10/08/2016
10436                            <1>       ; Read All Palette Registers
10437                            <1>       ; EDX = Address of 17-byte buffer
10438                            <1>       ;      to receive data
10439                            <1>
10440 00003768 89D7              <1>       mov    edi, edx
10441 0000376A 89E3              <1>       mov    ebx, esp
10442 0000376C 89DE              <1>       mov    esi, ebx
10443 0000376E 83EC14            <1>       sub    esp, 20
10444                            <1>
10445 00003771 B100              <1>       mov    cl, 0
10446                            <1> get_palette_loop:
10447 00003773 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10448 00003777 EC                <1>       in     al, dx
10449 00003778 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10450 0000377C 88C8              <1>       mov    al, cl
```

```
10451 0000377E EE                <1>       out    dx, al
10452 0000377F 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10453 00003783 EC                <1>       in     al, dx
10454 00003784 8803              <1>       mov    [ebx], al
10455 00003786 43                <1>       inc    ebx
10456 00003787 FEC1              <1>       inc    cl
10457 00003789 80F910            <1>       cmp    cl, 10h
10458 0000378C 75E5              <1>       jne    short get_palette_loop
10459 0000378E 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10460 00003792 EC                <1>       in     al, dx
10461 00003793 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10462 00003797 B011              <1>       mov    al, 11h
10463 00003799 EE                <1>       out    dx, al
10464 0000379A 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10465 0000379E EC                <1>       in     al, dx
10466 0000379F 8803              <1>       mov    [ebx], al
10467 000037A1 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10468 000037A5 EC                <1>       in     al, dx
10469 000037A6 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10470 000037AA B020              <1>       mov    al, 20h
10471 000037AC EE                <1>       out    dx, al
10472                            <1>       ; ifdef VBOX
10473 000037AD 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10474 000037B1 EC                <1>       in     al, dx
10475                            <1>       ; endif ; VBOX
10476                            <1>
10477 000037B2 B911000000        <1>       mov    ecx, 17 ; transfer (byte) count
10478                            <1>       ; ESI = source address in system space
10479                            <1>       ; EDI = user's buffer address
10480 000037B7 E868E20000        <1>       call   transfer_to_user_buffer
10481                            <1>
10482 000037BC 83C414            <1>       add    esp, 20
10483 000037BF E92DE3FFFF        <1>       jmp    VIDEO_RETURN
10484                            <1>
10485                            <1> set_single_dac_reg:
10486                            <1>       ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
10487                            <1>       ; 10/08/2016
10488                            <1>       ; Set One DAC Color Register
10489                            <1>       ; BX = color register to set (0-255)
10490                            <1>         ; CH = green value (00h-3Fh)
10491                            <1>         ; CL = blue value  (00h-3Fh)
10492                            <1>         ; DH = red value   (00h-3Fh)
10493                            <1>
10494                            <1>       ;push  dx
10495                            <1>       ; 12/04/2021
10496 000037C4 52                <1>       push   edx
10497 000037C5 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10498 000037C9 88D8              <1>       mov    al, bl
10499 000037CB EE                <1>       out    dx, al
10500                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
10501 000037CC 6642              <1>       inc    dx
10502                            <1>       ;pop   ax
10503                            <1>       ; 12/04/2021
10504 000037CE 58                <1>       pop    eax
10505 000037CF 88E0              <1>       mov    al, ah
10506 000037D1 EE                <1>       out    dx, al
10507 000037D2 88E8              <1>       mov    al, ch
10508 000037D4 EE                <1>       out    dx, al
10509 000037D5 88C8              <1>       mov    al, cl
10510 000037D7 EE                <1>       out    dx, al
10511 000037D8 E914E3FFFF        <1>       jmp    VIDEO_RETURN
10512                            <1>
10513                            <1> set_all_dac_reg:
10514                            <1>       ; 12/08/2016
10515                            <1>       ; 11/08/2016
10516                            <1>       ; 10/08/2016
10517                            <1>       ; Set a Block of DAC Color Register
10518                            <1>       ; BX = first DAC register to set (0-00FFh)
10519                            <1>       ; ECX = number of registers to set (0-00FFh)
10520                            <1>       ; EDX = addr of a table of R,G,B values
10521                            <1>       ;      (it will be CX*3 bytes long)
10522                            <1>
10523 000037DD 89D6              <1>       mov    esi, edx ; user buffer
10524 000037DF 89CA              <1>       mov    edx, ecx
10525 000037E1 66D1E1            <1>       shl    cx, 1 ; *2
10526 000037E4 01D1              <1>       add    ecx, edx ; ecx = 3*ecx
10527 000037E6 89E5              <1>       mov    ebp, esp
10528 000037E8 89EF              <1>       mov    edi, ebp
10529 000037EA 29CF              <1>       sub    edi, ecx
10530 000037EC 6683E7FC          <1>       and    di, 0FFFCh ; (dword alignment)
10531 000037F0 89FC              <1>       mov    esp, edi
10532 000037F2 E877E20000        <1>       call   transfer_from_user_buffer
10533                            <1>       ;jc    VIDEO_RETURN
10534                            <1>
10535 000037F7 89D1              <1>       mov    ecx, edx
10536 000037F9 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10537 000037FD 88D8              <1>       mov    al, bl
10538 000037FF EE                <1>       out    dx, al
10539 00003800 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
10540                            <1> set_dac_loop:
10541 00003804 8A07              <1>       mov    al, [edi]
10542 00003806 EE                <1>       out    dx, al
10543 00003807 47                <1>       inc    edi
10544 00003808 8A07              <1>       mov    al, [edi]
10545 0000380A EE                <1>       out    dx, al
10546 0000380B 47                <1>       inc    edi
10547 0000380C 8A07              <1>       mov    al, [edi]
10548 0000380E EE                <1>       out    dx, al
10549 0000380F 47                <1>       inc    edi
10550 00003810 6649              <1>       dec    cx
10551 00003812 75F0              <1>       jnz    short set_dac_loop
10552 00003814 89EC              <1>       mov    esp, ebp
10553 00003816 E9D6E2FFFF        <1>       jmp    VIDEO_RETURN
10554                            <1>
10555                            <1> select_video_dac_color_page:
```

```
10556                                  <1>       ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
10557                                  <1>       ; 10/08/2016
10558                                  <1>       ; DAC Color Paging Functions
10559                                  <1>       ; BL = 00H = select color paging mode
10560                                  <1>         ;       BH = paging mode
10561                                  <1>         ;            00h = 4 blocks of 64 registers
10562                                  <1>         ;            01h = 16 blocks of 16 registers
10563                                  <1>       ; BL = 01H = activate color page
10564                                  <1>         ;       BH = DAC color page number
10565                                  <1>         ;            00h-03h (4-page/64-reg mode)
10566                                  <1>         ;            00h-0Fh (16-page/16-reg mode)
10567                                  <1>
10568 0000381B 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10569 0000381F EC                     <1>       in     al, dx
10570 00003820 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10571 00003824 B010                   <1>       mov    al, 10h
10572 00003826 EE                     <1>       out    dx, al
10573 00003827 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10574 0000382B EC                     <1>       in     al, dx
10575 0000382C 80E301                 <1>       and    bl, 01h
10576 0000382F 750E                   <1>       jnz    short set_dac_page
10577 00003831 247F                   <1>       and    al, 07Fh
10578 00003833 C0E707                 <1>       shl    bh, 7
10579 00003836 08F8                   <1>       or     al, bh
10580 00003838 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10581 0000383C EE                     <1>       out    dx, al
10582 0000383D EB1B                   <1>       jmp    short set_actl_normal
10583                                  <1> set_dac_page:
10584                                  <1>       ;push  ax
10585                                  <1>       ; 12/04/2021
10586 0000383F 50                     <1>       push   eax
10587 00003840 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10588 00003844 EC                     <1>       in     al, dx
10589 00003845 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10590 00003849 B014                   <1>       mov    al, 14h
10591 0000384B EE                     <1>       out    dx, al
10592                                  <1>       ;pop   ax
10593                                  <1>       ; 12/04/2021
10594 0000384C 58                     <1>       pop    eax
10595 0000384D 2480                   <1>       and    al, 80h
10596 0000384F 7503                   <1>       jnz    short set_dac_16_page
10597 00003851 C0E702                 <1>       shl    bh, 2
10598                                  <1> set_dac_16_page:
10599 00003854 80E70F                 <1>       and    bh, 0Fh
10600 00003857 88F8                   <1>       mov    al, bh
10601 00003859 EE                     <1>       out    dx, al
10602                                  <1> set_actl_normal:
10603 0000385A B020                   <1>       mov    al, 20h
10604 0000385C EE                     <1>       out    dx, al
10605                                  <1>       ; ifdef VBOX
10606 0000385D 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10607 00003861 EC                     <1>       in     al, dx
10608                                  <1>       ; endif ; VBOX
10609 00003862 E98AE2FFFF             <1>       jmp    VIDEO_RETURN
10610                                  <1>
10611                                  <1> read_single_dac_reg:
10612                                  <1>       ; 10/08/2016
10613                                  <1>       ; Read One DAC Color Register
10614                                  <1>       ; INPUT:
10615                                  <1>       ; BX = color register to read (0-255)
10616                                  <1>       ; OUTPUT:
10617                                  <1>       ; CH = green value (00h-3Fh)
10618                                  <1>         ; CL = blue value  (00h-3Fh)
10619                                  <1>         ; DH = red value   (00h-3Fh)
10620                                  <1>
10621 00003867 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10622 0000386B 88D8                   <1>       mov    al, bl
10623 0000386D EE                     <1>       out    dx, al
10624 0000386E 66BAC903               <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
10625 00003872 EC                     <1>       in     al, dx
10626 00003873 88442415               <1>       mov    [esp+21], al ; dh
10627 00003877 EC                     <1>       in     al, dx
10628 00003878 88C5                   <1>       mov    ch, al
10629 0000387A EC                     <1>       in     al, dx
10630 0000387B 88C1                   <1>       mov    cl, al
10631 0000387D 66894C2410             <1>       mov    [esp+16], cx ; cx
10632 00003882 E96AE2FFFF             <1>       jmp    VIDEO_RETURN
10633                                  <1>
10634                                  <1> read_all_dac_reg:
10635                                  <1>       ; 12/08/2016
10636                                  <1>       ; 11/08/2016
10637                                  <1>       ; 10/08/2016
10638                                  <1>       ; Read a Block of DAC Color Registers
10639                                  <1>         ; BX = first DAC register to read (0-00FFh)
10640                                  <1>         ; ECX = number of registers to read (0-00FFh)
10641                                  <1>         ; EDX = addr of a buffer to hold R,G,B values
10642                                  <1>       ;      (CX*3 bytes long)
10643                                  <1>
10644 00003887 89D7                   <1>       mov    edi, edx ; user buffer
10645 00003889 89CA                   <1>       mov    edx, ecx
10646 0000388B 66D1E2                 <1>       shl    dx, 1 ; *2
10647 0000388E 01CA                   <1>       add    edx, ecx ; edx = 3*ecx
10648 00003890 89E5                   <1>       mov    ebp, esp
10649 00003892 89EE                   <1>       mov    esi, ebp
10650 00003894 29D6                   <1>       sub    esi, edx
10651 00003896 6683E6FC               <1>       and    si, 0FFFCh ; (dword alignment)
10652 0000389A 89F4                   <1>       mov    esp, esi
10653 0000389C 52                     <1>       push   edx ; 3*ecx
10654 0000389D 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10655 000038A1 88D8                   <1>       mov    al, bl
10656 000038A3 EE                     <1>       out    dx, al
10657 000038A4 66BAC903               <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
10658 000038A8 89F3                   <1>       mov    ebx, esi
10659                                  <1> read_dac_loop:
10660 000038AA EC                     <1>       in     al, dx
```

```
10661 000038AB 8803              <1>       mov    [ebx], al
10662 000038AD 43                <1>       inc    ebx
10663 000038AE EC                <1>       in     al, dx
10664 000038AF 8803              <1>       mov    [ebx], al
10665 000038B1 43                <1>       inc    ebx
10666 000038B2 EC                <1>       in     al, dx
10667 000038B3 8803              <1>       mov    [ebx], al
10668 000038B5 43                <1>       inc    ebx
10669 000038B6 6649              <1>       dec    cx
10670 000038B8 75F0              <1>       jnz    short read_dac_loop
10671 000038BA 59                <1>       pop    ecx ; 3*ecx
10672                            <1>       ; ECX = transfer (byte) count
10673                            <1>       ; ESI = source address in system space
10674                            <1>       ; EDI = user's buffer address
10675 000038BB E864E10000        <1>       call   transfer_to_user_buffer
10676 000038C0 89EC              <1>       mov    esp, ebp
10677 000038C2 E92AE2FFFF        <1>       jmp    VIDEO_RETURN
10678                            <1>
10679                            <1> set_pel_mask:
10680                            <1>       ; 10/08/2016
10681                            <1>       ; BL = mask value
10682 000038C7 66BAC603          <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
10683 000038CB 88D8              <1>       mov    al, bl
10684 000038CD EE                <1>       out    dx, al
10685 000038CE E91EE2FFFF        <1>       jmp    VIDEO_RETURN
10686                            <1>
10687                            <1> read_pel_mask:
10688                            <1>       ; 10/08/2016
10689                            <1>       ; Output: BL = mask value
10690 000038D3 66BAC603          <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
10691 000038D7 EC                <1>       in     al, dx
10692 000038D8 8844240C          <1>       mov    [esp+12], al ; bl
10693 000038DC E910E2FFFF        <1>       jmp    VIDEO_RETURN
10694                            <1>
10695                            <1> read_video_dac_state:
10696                            <1>       ; 10/08/2016
10697                            <1>       ; Query DAC Color Paging State
10698                            <1>       ; Output:
10699                            <1>       ; BH = current active DAC color page
10700                            <1>         ; BL = current active DAC paging mode
10701                            <1>
10702 000038E1 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10703 000038E5 EC                <1>       in     al, dx
10704 000038E6 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10705 000038EA B010              <1>       mov    al, 10h
10706 000038EC EE                <1>       out    dx, al
10707 000038ED 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10708 000038F1 EC                <1>       in     al, dx
10709 000038F2 88C3              <1>       mov    bl, al
10710 000038F4 C0EB07            <1>       shr    bl, 7
10711 000038F7 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10712 000038FB EC                <1>       in     al, dx
10713 000038FC 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10714 00003900 B014              <1>       mov    al, 14h
10715 00003902 EE                <1>       out    dx, al
10716 00003903 66BAC103          <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10717 00003907 EC                <1>       in     al, dx
10718 00003908 88C7              <1>       mov    bh, al
10719 0000390A 80E70F            <1>       and    bh, 0Fh
10720 0000390D F6C301            <1>       test   bl, 01
10721 00003910 7503              <1>       jnz    short get_dac_16_page
10722 00003912 C0EF02            <1>       shr    bh, 2
10723                            <1> get_dac_16_page:
10724 00003915 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10725 00003919 EC                <1>       in     al, dx
10726 0000391A 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10727 0000391E B020              <1>       mov    al, 20h
10728 00003920 EE                <1>       out    dx, al
10729                            <1>       ; ifdef VBOX
10730 00003921 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10731 00003925 EC                <1>       in     al, dx
10732                            <1>       ; endif ; VBOX
10733 00003926 66895C240C        <1>       mov    [esp+12], bx ; bx
10734 0000392B E9C1E1FFFF        <1>       jmp    VIDEO_RETURN
10735                            <1>
10736                            <1> ; 23/11/2020 - TRDOS 386 v2.0.3
10737                            <1> ; VBE 2 BOCHS/QEMU emulator extensions
10738                            <1> ;     for TRDOS 386 v2 kernel (video bios)
10739                            <1>
10740                            <1> ; BOCH/QEMU VBE2 VGA BIOS code
10741                            <1> ;     by Jeroen Janssen (2002)
10742                            <1> ;     by Volker Rupper (2003-2020)
10743                            <1> ; vbe.c (02/01/2020)
10744                            <1>
10745                            <1> ; vbe.h (02/01/2020)
10746                            <1>
10747                            <1> VBE_DISPI_BANK_ADDRESS    equ    0A0000h
10748                            <1> VBE_DISPI_BANK_SIZE_KB    equ    64
10749                            <1>
10750                            <1> VBE_DISPI_MAX_XRES equ    2560
10751                            <1> VBE_DISPI_MAX_YRES equ    1600
10752                            <1>
10753                            <1> VBE_DISPI_IOPORT_INDEX    equ    01CEh
10754                            <1> VBE_DISPI_IOPORT_DATA     equ    01CFh
10755                            <1>
10756                            <1> VBE_DISPI_INDEX_ID equ    00h
10757                            <1> VBE_DISPI_INDEX_XRES      equ    01h
10758                            <1> VBE_DISPI_INDEX_YRES      equ    02h
10759                            <1> VBE_DISPI_INDEX_BPP       equ    03h
10760                            <1> VBE_DISPI_INDEX_ENABLE    equ    04h
10761                            <1> VBE_DISPI_INDEX_BANK      equ    05h
10762                            <1> VBE_DISPI_INDEX_VIRT_WIDTH equ  06h
10763                            <1> VBE_DISPI_INDEX_VIRT_HEIGHT equ 07h
10764                            <1> VBE_DISPI_INDEX_X_OFFSET equ    08h
10765                            <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>
00003930 55                     <1>     push   ebp ; ***
00003931 56                     <1>     push   esi ; ****
                                <1>
```

```
10871 00003932 31FF               <1>       xor   edi, edi  ; mov edi, 0
10872                             <1>
10873 00003934 803D[52090000]03   <1>       cmp   byte [vbe3], 3
10874 0000393B 7221               <1>       jb    short _vbe_rmi_1
10875                             <1>
10876                             <1>       ;sub   edi, edi  ; 0 = kernel call (sign)
10877                             <1>       ;      ; no transfer to user's buffer
10878                             <1>
10879                             <1>       ; cx = Video mode (for 4F01h, with LFB flag)
10880                             <1>
10881 0000393D 66B8014F           <1>       mov   ax, 4F01h
10882                             <1>
10883 00003941 E857DFFFFF         <1>       call  _vbe3_pmfn_return_mode_info
10884                             <1>
10885 00003946 6683F84F           <1>       cmp   ax, 004Fh
10886 0000394A 7533               <1>       jne   short _vbe_rmi_2 ; fail
10887                             <1>
10888                             <1>       ; 15/12/2020
10889                             <1>       ; cx = vbe video mode
10890 0000394C 80E501             <1>       and   ch, 01h ; clear LFB flag
10891 0000394F BEFE7B0900         <1>       mov   esi, VBE3MODEINFOBLOCK - 2
10892 00003954 66890E             <1>       mov   [esi], cx ; MODEINFO.mode
10893 00003957 E8A6000000         <1>       call  set_lfbinfo_table
10894 0000395C EB22               <1>       jmp   short _vbe_rmi_3  ; cf = 0
10895                             <1> _vbe_rmi_1:
10896 0000395E 803D[52090000]02   <1>       cmp   byte [vbe3], 2
10897 00003965 7219               <1>       jb    short _vbe_rmi_3 ; cf = 1
10898 00003967 A0[53090000]       <1>       mov   al, [vbe2bios] ; 0C0h-0C5h for emu (*)
10899 0000396C 3CC0               <1>       cmp   al, 0C0h ; BOCHS/QEMU/VIRTUALBOX (*) ?
10900 0000396E 7210               <1>       jb    short _vbe_rmi_3  ; cf = 1
10901 00003970 3CC5               <1>       cmp   al, 0C5h ; (*)
10902 00003972 770B               <1>       ja    short _vbe_rmi_2 ; unknown vbios !?
10903                             <1>
10904                             <1>       ;xor   edi, edi  ; 0 = kernel call (sign)
10905                             <1>       ;      ; no transfer to user's buffer
10906                             <1>
10907                             <1>       ;mov   ax, 4F01h
10908                             <1>
10909                             <1>       ; cx = Video mode (for 4F01h, with LFB flag)
10910                             <1>
10911 00003974 E80A000000         <1>       call  vbe_biosfn_return_mode_info
10912 00003979 6683F84F           <1>       cmp   ax, 004Fh ; successful ?
10913 0000397D 7401               <1>       je    short _vbe_rmi_3  ; cf = 0
10914                             <1> _vbe_rmi_2:
10915 0000397F F9                 <1>       stc
10916                             <1>       ; cf = 1
10917                             <1> _vbe_rmi_3:
10918 00003980 5E                 <1>       pop   esi ; ****
10919 00003981 5D                 <1>       pop   ebp ; ***
10920                             <1>
10921                             <1>       ;pop  es  ; **
10922                             <1>       ;pop  ss  ; *
10923                             <1>
10924 00003982 C3                 <1>       retn
10925                             <1>
10926                             <1>
10927                             <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
10928                             <1> ; * ------------------------------------------------------
10929                             <1> ; * Function 01h - Return VBE Mode Information
10930                             <1> ; * ------------------------------------------------------
10931                             <1> ; * Input:
10932                             <1> ; *         AX = 4F01h
10933                             <1> ; *         CX = Mode number
10934                             <1> ; *    (ES:DI) EDI = Pointer to ModeInfoBlock structure
10935                             <1> ; * Output:
10936                             <1> ; *         AX = VBE Return Status
10937                             <1> ; *
10938                             <1> ; *------------------------------------------------------
10939                             <1> ; *
10940                             <1>
10941                             <1> vbe_biosfn_return_mode_info:
10942                             <1>       ; 15/12/2020
10943                             <1>       ; 14/12/2020
10944                             <1>       ; 12/12/2020
10945                             <1>       ; 11/12/2020 (TRDOS 386 v2.0.3)
10946                             <1>       ;
10947                             <1>       ; Input:
10948                             <1>       ;      cx = video (bios) mode
10949                             <1>       ;      edi = ModeInfoBlock buffer address
10950                             <1>       ;           (in user's memory space)
10951                             <1>       ;      (ax = 4F01h)
10952                             <1>       ; Output:
10953                             <1>       ;      ax = 004Fh (successful)
10954                             <1>       ;      ah > 0 -> error
10955                             <1>       ;
10956                             <1>       ; Modified registers: esi
10957                             <1>
10958                             <1>       ;;push ds  ; *
10959                             <1>       ;;push es  ; **
10960                             <1>       ;;push ebp ; ***
10961                             <1>       ;;push esi ; ****
10962                             <1>
10963 00003983 F6C501             <1>       test  ch, 1
10964 00003986 7505               <1>       jnz   short vbe_rmi_1
10965                             <1>
10966                             <1>       ; mode number < 100h
10967                             <1>       ; CGA/VGA mode is not proper this VBE function
10968                             <1>
10969 00003988 29C0               <1>       sub   eax, eax
10970                             <1> vbe_rmi_0:
10971                             <1>       ;mov   ax, 0100h  ; Function is not supported
10972 0000398A B401               <1>       mov   ah, 1
10973 0000398C C3                 <1>       retn
10974                             <1> vbe_rmi_1:
10975 0000398D 52                 <1>       push  edx ; *****
```

```
10976 0000398E 51                  <1>       push   ecx ; ******
10977 0000398F 53                  <1>       push   ebx ; *******
10978 00003990 57                  <1>       push   edi ; ********
10979                              <1>
10980                              <1>       ; 14/12/2020
10981 00003991 89CB                <1>       mov    ebx, ecx
10982                              <1>
10983                              <1>       ;xor   eax, eax
10984 00003993 80E7C1              <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
10985 00003996 883D[1D120300]      <1>       mov    [vbe_mode_x], bh
10986                              <1>       ;and   bx, 1FFh
10987 0000399C 80E701              <1>       and    bh, 1
10988                              <1>       ;mov   bh, 1
10989                              <1>
10990                              <1>       ; Alternative 2 (instead of 'Mode_info_find_mode')
10991 0000399F E88A060000          <1>       call   set_mode_info_list ; (alternative 2)
10992                              <1>
10993                              <1>       ; eax = 0
10994                              <1>
10995                              <1>       ;mov   bx, [esi] ; mode
10996                              <1>
10997                              <1>       ; Alternative 1 (instead of 'set_mode_info_list')
10998                              <1>       ;call  mode_info_find_mode ; (alternative 1)
10999                              <1>
11000 000039A4 09F6                <1>       or     esi, esi
11001                              <1>       ; 14/12/2020
11002 000039A6 744D                <1>       jz     short vbe_rmi_4     ; VBE mode number is wrong
11003                              <1>                                  ; or it is not supported
11004                              <1>
11005                              <1>       ; 15/12/2020
11006                              <1>       ;mov   bx, [esi] ; mode
11007                              <1>
11008                              <1>       ; 12/12/2020
11009                              <1>       ;call  set_lfbinfo_table
11010                              <1>
11011 000039A8 F605[1D120300]40    <1>       test   byte [vbe_mode_x], 40h ; LFB model ?
11012 000039AF 7404                <1>       jz     short vbe_rmi_2
11013                              <1>
11014 000039B1 C6461C01            <1>       mov    byte [esi+MODEINFO.NumberOfBanks], 1
11015                              <1> vbe_rmi_2:
11016                              <1>       ; (vbe.c, 02/01/2020, vruppert)
11017                              <1>       ; 11/12/2020 (Erdogan Tan, video.s)
11018                              <1>       ; Bochs Graphics Adapter
11019                              <1>       ; vendor_id: 1111h, device id: 1234h
11020                              <1>
11021 000039B5 E855070000          <1>       call   pci_get_lfb_addr
11022                              <1>       ;or    eax, eax
11023 000039BA 7404                <1>       jz     short vbe_rmi_3
11024                              <1>       ; zf = 0, ax > 0 (high word of LFB address)
11025                              <1>       ; set/change LFB address in MODEINFO structure
11026 000039BC 6689462C            <1>       mov    [esi+MODEINFO.PhysBasePtr+2], ax
11027                              <1>       ; 12/12/2020
11028                              <1>       ;mov   [edi+LFBINFO.LFB_addr+2], ax
11029                              <1> vbe_rmi_3:
11030                              <1>       ;test  byte [esi+MODEINFO.WinAAttributes], 1
11031                              <1>       ;            ; VBE_WINDOW_ATTRIBUTE_RELOCATABLE = 1
11032                              <1>         ;jz  short vbe_rmi_4
11033                              <1>       ;; 11/12/2020
11034                              <1>       ;; In fact, this is far call address in (Bochs/BGA) Video Bios
11035                              <1>       ;; Direct user access to kernel subroutines is not possible
11036                              <1>       ;; in TRDOS 386. Also, TRDOS 386 kernel will support only LFB.
11037                              <1>       ;; Bank select may be a seperate sysvideo function in future
11038                              <1>       ;; (if it will be required).
11039                              <1>       ;mov   dword [esi+MODEINFO.WinFuncPtr], dispi_set_bank_farcall
11040                              <1> ;vbe_rmi_4:
11041                              <1>       ; 12/12/2020
11042 000039C0 E83D000000          <1>       call   set_lfbinfo_table
11043                              <1>
11044                              <1>       ; 11/12/2020
11045                              <1>       ; copy 68 bytes of MODE_INFO_LIST to user
11046                              <1>
11047 000039C5 8B3C24              <1>       mov    edi, [esp]  ; user's buffer address
11048                              <1>       ; 12/12/2020
11049 000039C8 09FF                <1>       or     edi, edi ; 0 = kernel call
11050                              <1>                       ; (call from '_vbe_biosfn_return_mode_info')
11051 000039CA 7432                <1>       jz     short vbe_rmi_6
11052                              <1>
11053                              <1>       ; 15/12/2020
11054                              <1>       ; prepare 256 bytes MODEINFO buffer at VBE3MODEINFOBLOCK
11055                              <1>       ; and then, copy buffer conttent to user's buffer
11056 000039CC 57                  <1>       push   edi
11057 000039CD BE[3C120300]        <1>       mov    esi, MODE_INFO_LIST + 2   ; MODEINFO.ModeAttributes
11058 000039D2 BF007C0900          <1>       mov    edi, VBE3MODEINFOBLOCK
11059 000039D7 B910000000          <1>       mov    ecx, 66/4  ; 66 bytes
11060 000039DC F3A5                <1>       rep    movsd
11061 000039DE 31C0                <1>       xor    eax, eax
11062 000039E0 B12F                <1>       mov    cl, (256-68)/4 ; 188 bytes
11063 000039E2 F3AB                <1>       rep    stosd
11064 000039E4 66AB                <1>       stosw  ; 2 bytes
11065 000039E6 5F                  <1>       pop    edi
11066 000039E7 BE007C0900          <1>       mov    esi, VBE3MODEINFOBLOCK
11067                              <1>       ;mov   cx, 256
11068 000039EC FEC5                <1>       inc    ch ; cx = 256
11069 000039EE E831E00000          <1>       call   transfer_to_user_buffer
11070 000039F3 7309                <1>       jnc    short vbe_rmi_5
11071                              <1> vbe_rmi_4:
11072                              <1>       ;mov   eax, 014Fh ; fail/error
11073 000039F5 31C0                <1>       xor    eax, eax
11074 000039F7 B401                <1>       mov    ah, 01h
11075                              <1>       ;jmp   short vbe_rmi_6
11076 000039F9 E981000000          <1>       jmp    vbe_sm_ret1 ; 11/12/2020
11077                              <1> vbe_rmi_5:
11078                              <1>       ; 256 bytes of MODEINFO have been transferred to user
11079                              <1>       ;mov   eax, 4Fh ; succesfull
11080                              <1> vbe_rmi_6: ; 12/12/2020
```

```
11081 000039FE 31C0               <1>       xor    eax, eax
11082                             <1> ;vbe_rmi_6:
11083 00003A00 EB7D               <1>       jmp    vbe_sm_ret1 ; 11/12/2020
11084                             <1>
11085                             <1>       ;pop   edi ; ********
11086                             <1>       ;pop   ebx ; *******
11087                             <1>       ;pop   ecx ; ******
11088                             <1>       ;pop   edx ; *****
11089                             <1>
11090                             <1>       ;;pop  esi ; ****
11091                             <1>       ;;pop  ebp ; ***
11092                             <1>       ;;pop  es  ; **
11093                             <1>       ;;pop  ds  ; *
11094                             <1>
11095                             <1>       ;retn
11096                             <1>
11097                             <1> set_lfbinfo_table:
11098                             <1>       ; 19/12/2020
11099                             <1>       ; 11/12/2020
11100                             <1>       ; Set/Fill LFBINFO structure/table
11101                             <1>       ;
11102                             <1>       ; Input:
11103                             <1>       ;      esi = Mode info list address
11104                             <1>       ; Output:
11105                             <1>       ;      LFB_Info address is filled with LFBINFO
11106                             <1>       ;      edi = LFB_Info address
11107                             <1>       ;
11108                             <1>       ; Modified registers: eax, edx (=0), edi
11109                             <1>
11110 00003A02 BF[2A120300]       <1>       mov    edi, LFB_Info
11111 00003A07 8B462A             <1>       mov    eax, [esi+MODEINFO.PhysBasePtr]
11112 00003A0A 894702             <1>       mov    [edi+LFBINFO.LFB_addr], eax ; LFB address
11113                             <1>       ;mov   ax, [esi+MODEINFO.mode]
11114 00003A0D 668B06             <1>       mov    ax, [esi]
11115 00003A10 668907             <1>       mov    [edi+LFBINFO.mode],ax
11116 00003A13 8A461B             <1>       mov    al, [esi+MODEINFO.BitsPerPixel]
11117 00003A16 88470E             <1>       mov    [edi+LFBINFO.bpp], al
11118 00003A19 29C0               <1>       sub    eax, eax
11119 00003A1B 668B4614           <1>       mov    ax, [esi+MODEINFO.XResolution]
11120 00003A1F 6689470A           <1>       mov    [edi+LFBINFO.X_res], ax
11121 00003A23 89C2               <1>       mov    edx, eax ; 19/12/2020
11122 00003A25 668B4616           <1>       mov    ax, [esi+MODEINFO.YResolution]
11123 00003A29 6689470C           <1>       mov    [edi+LFBINFO.Y_res], ax
11124                             <1>       ; eax = Y_res ; screen height
11125                             <1>       ; 19/12/2020
11126 00003A2D F7E2               <1>       mul    edx ; X_res*Y_res
11127                             <1>       ; edx = 0
11128 00003A2F 8A570E             <1>       mov    dl, [edi+LFBINFO.bpp]
11129                             <1>       ; Note:
11130                             <1>       ; Bits per pixel may be 8,16,24,32 for TRDOS 386 v2.
11131                             <1>       ; (4 bits for pixel is not used for VESA modes here)
11132 00003A32 C0EA03             <1>       shr    dl, 3 ; convert bits to byte
11133 00003A35 F7E2               <1>       mul    edx
11134                             <1>       ; eax = screen/page/buffer size in bytes
11135 00003A37 894706             <1>       mov    [edi+LFBINFO.LFB_size], eax
11136                             <1>       ; edx = 0
11137                             <1>       ; clear reserved byte in LFBINFO structure/table
11138 00003A3A 88570F             <1>       mov    [edi+LFBINFO.reserved], dl ; not necessary
11139 00003A3D C3                 <1>       retn
11140                             <1>
11141                             <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
11142                             <1> ; * ------------------------------------------------------
11143                             <1> ; * Function 02h - Set VBE Mode
11144                             <1> ; * ------------------------------------------------------
11145                             <1> ; * Input:
11146                             <1> ; *         AX = 4F02h
11147                             <1> ; *         BX = Desired Mode to set
11148                             <1> ; * Output:
11149                             <1> ; *         AX = VBE Return Status
11150                             <1> ; *
11151                             <1> ; *------------------------------------------------------
11152                             <1> ; *
11153                             <1>
11154                             <1> vbe_biosfn_set_mode:
11155                             <1>       ; 07/03/2021
11156                             <1>       ; 12/12/2020
11157                             <1>       ; 11/12/2020 (LFBINFO table for VESA VBE modes)
11158                             <1>       ; 27/11/2020
11159                             <1>       ; 25/11/2020
11160                             <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
11161                             <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
11162                             <1>       ;
11163                             <1>       ; Input:
11164                             <1>       ;      bx = video (bios) mode
11165                             <1>       ;      ax = 4F02h
11166                             <1>       ; Output:
11167                             <1>       ;      ax = 004Fh (successful)
11168                             <1>       ;      ah > 0 -> error
11169                             <1>       ;
11170                             <1>       ; Modified registers: esi
11171                             <1>
11172                             <1>       ; 27/11/2020
11173                             <1>
11174                             <1>       ;;push ds  ; *
11175                             <1>       ;;push es  ; **
11176                             <1>       ;;push ebp ; ***
11177                             <1>       ;;push esi ; ****
11178                             <1>
11179                             <1>       ; 11/12/2020
11180 00003A3E 52                 <1>       push   edx ; *****
11181 00003A3F 51                 <1>       push   ecx ; ******
11182 00003A40 53                 <1>       push   ebx ; *******
11183 00003A41 57                 <1>       push   edi ; ********
11184                             <1>
11185                             <1>       ;xor   eax, eax
```

```
11186 00003A42 80E7C1              <1>       and   bh, 0C1h ; use bit 15, 14, 8 only (for bh)
11187 00003A45 883D[1D120300]      <1>       mov   [vbe_mode_x], bh
11188 00003A4B 80E701              <1>       and   bh, 1
11189 00003A4E 753C                <1>       jnz   short vbe_sm_3  ; VESA VBE mode
11190                              <1>
11191                              <1>       ;;test bx, 4000h ; VBE_MODE_LINEAR_FRAME_BUFFER
11192                              <1>       ;test  bh, 40h
11193                              <1>       ;jz    short vbe_sm_0
11194                              <1>       ;; lfb_flag
11195                              <1>       ;mov   al, 40h      ; VBE_DISPI_LFB_ENABLED
11196                              <1> vbe_sm_0:
11197                              <1>       ; 27/11/2020
11198 00003A50 B080                <1>       mov   al, 80h
11199                              <1>       ;test  bh, 80h ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
11200                              <1>       ;jnz   short vbe_sm_1 ; no_clear
11201                              <1>       ;; clear
11202                              <1>       ;sub   al, al ; 0
11203 00003A52 8405[1D120300]      <1>       test   [vbe_mode_x], al ; 80h
11204 00003A58 7402                <1>       jz     short vbe_sm_1 ; clear display memory
11205                              <1>       ; no_clear
11206 00003A5A 08C3                <1>       or     bl, al ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
11207                              <1> vbe_sm_1:
11208                              <1>       ; check non vesa mode
11209                              <1>       ;;cmp  bx, 100h ; VBE_MODE_VESA_DEFINED
11210                              <1>       ;;jna  short vbe_sm_2
11211                              <1>       ;and   bh, 1
11212                              <1>       ;jnz   short vbe_sm_3
11213                              <1>
11214                              <1>       ; BX <= 1FFh
11215                              <1>
11216                              <1>       ; 27/11/2020
11217                              <1>       ;or    bl, al ; al = 80h if no_clear option is set
11218                              <1>       ;             ; al = 0 if no_clear option is not set
11219                              <1>
11220                              <1>       ; 25/11/2020
11221                              <1>       ; VBE DISPI will be disabled in 'biosfn_set_video_mode'
11222                              <1>
11223                              <1>       ;xor   al, al ; 0 ; VBE_DISPI_DISABLED
11224                              <1>       ;call  dispi_set_enable
11225                              <1>
11226                              <1>       ; call the vgabios in order to set the video mode
11227                              <1>       ; this allows for going back to textmode with a VBE call
11228                              <1>       ; (some applications expect that to work)
11229                              <1>
11230                              <1>       ;and   bx, 0FFh
11231                              <1>
11232                              <1>       ; 27/11/2020
11233                              <1> biosfn_set_video_mode:
11234                              <1>       ; _call: call subroutine
11235                              <1>       ; 26/11/2020 (TRDOS 386 v2.0.3)
11236                              <1>       ; (ref: vgabios.c, 02/01/2020, vruppert)
11237                              <1>       ; Input:
11238                              <1>       ;     bl = VGA video (bios) mode
11239                              <1>       ; Output:
11240                              <1>       ;     cf = 1 -> error
11241                              <1>       ;     cf = 0 -> ok
11242                              <1>       ;
11243                              <1>       ; Modified registers: esi
11244                              <1>
11245                              <1>       ; 'dispi_set_enable(VBE_DISPI_DISABLED);'
11246                              <1>
11247                              <1>       ;mov   ax, 0 ; VBE_DISPI_DISABLED
11248 00003A5C 31C0                <1>       xor    eax, eax ; 0
11249 00003A5E E8A3040000          <1>       call   dispi_set_enable
11250                              <1>
11251 00003A63 88D8                <1>       mov    al, bl
11252                              <1>       ;jmp   _set_mode ; (in 'biosfn_set_video_mode' sub)
11253 00003A65 E898E0FFFF          <1>       call   _set_mode ; will return with cf=1 only if
11254                              <1>                        ; desired mode is not implemented
11255                              <1>       ; _retn: return from subroutine
11256 00003A6A 721A                <1>       jc     short vbe_sm_2 ; 25/11/2020
11257                              <1>
11258                              <1>       ; 26/11/2020
11259 00003A6C 31C0                <1>       xor    eax, eax
11260 00003A6E A0[9A6E0000]        <1>       mov    al, [CRT_MODE]
11261                              <1>       ; 27/11/2020
11262 00003A73 8A25[87950100]      <1>       mov    ah, [noclearmem] ; 80h or 0
11263                              <1>       ;and   ah 80h
11264 00003A79 66A3[1E120300]      <1>       mov    [video_mode], ax ; bit 15 = no_clear flag
11265                              <1>                               ; bit 14 = 0 (not LFB model)
11266                              <1> vbe_sm_ret1:
11267                              <1>       ; 11/12/2020
11268                              <1>       ; (vbe_rmi_4 and vbe_rmi_6 jump here)
11269                              <1>       ; 27/11/2020
11270 00003A7F B04F                <1>       mov    al, 4Fh ; Function call successful
11271                              <1>       ; eax = 004Fh
11272                              <1> vbe_sm_ret2:
11273                              <1>       ; 11/12/2020
11274 00003A81 5F                  <1>       pop    edi ; ********
11275 00003A82 5B                  <1>       pop    ebx ; *******
11276 00003A83 59                  <1>       pop    ecx ; ******
11277 00003A84 5A                  <1>       pop    edx ; *****
11278                              <1>
11279                              <1>       ;;pop  esi ; ****
11280                              <1>       ;;pop  ebp ; ***
11281                              <1>       ;;pop  es  ; **
11282                              <1>       ;;pop  ds  ; *
11283                              <1>
11284 00003A85 C3                  <1>       retn
11285                              <1>
11286                              <1> vbe_sm_2:
11287                              <1>       ;mov   ax, 0100h ; Function is not supported
11288                              <1>       ; 27/11/2020
11289 00003A86 31C0                <1>       xor    eax, eax
11290 00003A88 B401                <1>       mov    ah, 01h
```

```
11291                                  <1>       ; eax = 0100h
11292                                  <1>       ;retn
11293 00003A8A EBF5                    <1>       jmp    short vbe_sm_ret2
11294                                  <1>
11295                                  <1> vbe_sm_3:
11296                                  <1>       ; 12/12/2020
11297                                  <1>       ; check current mode, if it is 03h
11298                                  <1>       ; save page contents and cursor positions
11299 00003A8C 803D[9A6E0000]03        <1>       cmp    byte [CRT_MODE], 03h
11300                                  <1>       ;jne   short vbe_sm_0
11301 00003A93 7505                    <1>       jne    short vbe_sm_4 ; 07/03/2021
11302 00003A95 E81BE3FFFF              <1>       call   save_mode3_multiscreen
11303                                  <1>       ; set current mode to extended (SVGA) mode
11304                                  <1>       ;mov   byte [CRT_MODE], 0FFh ; VESA VBE mode
11305                                  <1> vbe_sm_4:
11306                                  <1>       ; 27/11/2020
11307                                  <1>       ; bx = mode (bit 0 to 8)
11308                                  <1>
11309                                  <1>       ; 25/11/2020
11310                                  <1>
11311                                  <1>       ; Alternative 2 (instead of 'Mode_info_find_mode')
11312                                  <1>       ;push  edi
11313 00003A9A E88F050000              <1>       call   set_mode_info_list ; (alternative 2)
11314                                  <1>       ;pop   edi
11315                                  <1>
11316                                  <1>       ;mov   bx, [esi] ; mode
11317                                  <1>
11318                                  <1>       ; Alternative 1 (instead of 'set_mode_info_list')
11319                                  <1>       ;call  mode_info_find_mode ; (alternative 1)
11320                                  <1>
11321 00003A9F 09F6                    <1>       or     esi, esi
11322 00003AA1 74E3                    <1>       jz     short vbe_sm_2 ; VBE mode number is wrong
11323                                  <1>                             ; or it is not supported
11324                                  <1>
11325                                  <1>       ; 11/12/2020
11326 00003AA3 668B1E                  <1>       mov    bx, [esi] ; mode
11327                                  <1>
11328                                  <1>       ; 27/11/2020
11329 00003AA6 0A3D[1D120300]          <1>       or     bh, [vbe_mode_x]
11330                                  <1>
11331                                  <1>       ; save VESA VBE mode
11332 00003AAC 66891D[1E120300]        <1>       mov    [video_mode], bx
11333                                  <1>              ; 27/11/2020
11334                                  <1>              ; bit 0 to 8 = VESA VBE mode
11335                                  <1>              ; bit 9 to 13 = 0 (bit 0 to 13 = mode)
11336                                  <1>              ; bit 14 = Linear/Flat Frame Buffer flag
11337                                  <1>              ; bit 15 = 'memory not cleared
11338                                  <1>              ;         at last mode set' flag
11339                                  <1>
11340                                  <1>       ; first disable current mode
11341                                  <1>       ; (when switching between vesa modes)
11342                                  <1>       ; 'dispi_set_enable(VBE_DISPI_DISABLED);'
11343                                  <1>
11344                                  <1>       ;mov   ax, VBE_DISPI_DISABLED ; 0
11345 00003AB3 29C0                    <1>       sub    eax, eax ; 0
11346                                  <1>
11347 00003AB5 E84C040000              <1>       call   dispi_set_enable
11348                                  <1>
11349                                  <1>       ; 11/12/2020
11350 00003ABA 8A461B                  <1>       mov    al, [esi+MODEINFO.BitsPerPixel]
11351                                  <1>       ; ah = 0
11352                                  <1>
11353                                  <1>       ;cmp   byte [esi+MODEINFO.BitsPerPixel], 8
11354 00003ABD 3C08                    <1>       cmp    al, 8
11355 00003ABF 750B                    <1>       jne    short vbe_sm_5
11356                                  <1>
11357                                  <1>       ; 11/12/2020
11358                                  <1>       ;push  edi
11359 00003AC1 50                      <1>       push   eax
11360                                  <1>       ; 'load_dac_palette(3);'
11361 00003AC2 56                      <1>       push   esi
11362 00003AC3 B403                    <1>       mov    ah, 3  ; palette3, 256 colors
11363 00003AC5 E83AF1FFFF              <1>       call   load_dac_palette
11364 00003ACA 5E                      <1>       pop    esi
11365                                  <1>       ; 11/12/2020
11366 00003ACB 58                      <1>       pop    eax
11367                                  <1>       ;pop   edi
11368                                  <1> vbe_sm_5:
11369                                  <1>       ;'dispi_set_bpp(cur_info->info.BitsPerPixel);'
11370                                  <1>       ; 11/12/2020 (al = bits per pixel, ah = 0)
11371                                  <1>       ;xor   ah, ah
11372                                  <1>       ;mov   al, [esi+MODEINFO.BitsPerPixel]
11373 00003ACC E849040000              <1>       call   dispi_set_bpp
11374                                  <1>         ;'dispi_set_xres(cur_info->info.XResolution);'
11375 00003AD1 668B4614                <1>       mov    ax, [esi+MODEINFO.XResolution]
11376 00003AD5 E846040000              <1>       call   dispi_set_xres
11377                                  <1>         ;'dispi_set_yres(cur_info->info.YResolution);'
11378 00003ADA 668B4616                <1>       mov    ax, [esi+MODEINFO.YResolution]
11379 00003ADE E843040000              <1>       call   dispi_set_yres
11380                                  <1>
11381                                  <1>       ;'dispi_set_bank(0);'
11382                                  <1>       ;xor   ax, ax
11383 00003AE3 31C0                    <1>       xor    eax, eax ; 0
11384 00003AE5 E842040000              <1>       call   dispi_set_bank
11385                                  <1>         ;'dispi_set_enable(VBE_DISPI_ENABLED|no_clear|lfb_flag);'
11386                                  <1>       ;mov   ax, di
11387                                  <1>       ; ah = 0 ; 27/11/2020
11388 00003AEA A0[1D120300]            <1>       mov    al, [vbe_mode_x] ; restore VBE mode bit 14 & 15
11389 00003AEF 0C01                    <1>       or     al, 1 ; VBE_DISPI_ENABLED
11390 00003AF1 E810040000              <1>       call   dispi_set_enable
11391                                  <1>
11392                                  <1>         ; 'vga_compat_setup();'
11393 00003AF6 E846040000              <1>       call   vga_compat_setup
11394                                  <1>
11395                                  <1>       ; 11/12/2020
```

```
11396 00003AFB E802FFFFFF          <1>       call   set_lfbinfo_table
11397                              <1>
11398                              <1>       ; 26/11/2020
11399 00003B00 31C0                <1>       xor    eax, eax
11400 00003B02 FEC8                <1>       dec    al
11401 00003B04 A2[9A6E0000]        <1>       mov    [CRT_MODE], al ; 0FFh = VESA VBE mode sign
11402                              <1>
11403                              <1>       ; 27/11/2020
11404 00003B09 E971FFFFFF          <1>       jmp    vbe_sm_ret1 ; Function call successful
11405                              <1>
11406                              <1>       ; 27/11/2020
11407                              <1>       ;mov   al, 4Fh
11408                              <1>       ;      ; eax = 004Fh = Function call successful
11409                              <1>       ;jmp   short vbe_sm_ret2
11410                              <1>
11411                              <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
11412                              <1> ; * ------------------------------------------------------
11413                              <1> ; * Function 03h - Return Current VBE Mode
11414                              <1> ; * ------------------------------------------------------
11415                              <1> ; * Input:
11416                              <1> ; *          AX = 4F03h
11417                              <1> ; * Output:
11418                              <1> ; *          AX = VBE Return Status
11419                              <1> ; *          BX = Current VBE Mode
11420                              <1> ; *
11421                              <1> ; *------------------------------------------------------
11422                              <1> ; *
11423                              <1>
11424                              <1> vbe_biosfn_return_current_mode:
11425                              <1>       ; 11/12/2020
11426                              <1>       ; 27/11/2020 (TRDOS 386 v2.0.3)
11427                              <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
11428                              <1>       ;
11429                              <1>       ; Input:
11430                              <1>       ;      none
11431                              <1>       ; Output:
11432                              <1>       ;      ax = 004Fh (successful)
11433                              <1>       ;      ah > 0 -> error
11434                              <1>       ;      bx = current video (bios) mode (if ah = 0)
11435                              <1>       ;
11436                              <1>       ; Modified registers: eax, ebx
11437                              <1>
11438                              <1>       ; 27/11/2020
11439                              <1>
11440                              <1>       ;;push ds  ; *
11441                              <1>       ;;push es  ; **
11442                              <1>       ;;push ebp ; ***
11443                              <1>       ;;push esi ; ****
11444                              <1>
11445                              <1>       ;push  edx ; *****
11446                              <1>
11447                              <1>       ; (vbe.c)
11448                              <1>       ;call  dispi_get_enable
11449                              <1>       ;      ; ax = vbe display interface status
11450                              <1>       ;and   al, 1 ; VBE_DISPI_ENABLED
11451                              <1>       ;jnz   short vbe_gm_1  ; VBE graphics mode
11452                              <1>
11453 00003B0E A0[9A6E0000]        <1>       mov    al, [CRT_MODE] ; current cga/vga mode
11454 00003B13 3CFF                <1>       cmp    al, 0FFh ; VBE extension signature
11455 00003B15 720E                <1>       jb     short vbe_gm_1 ; get CGA/VGA mode
11456                              <1>
11457                              <1>       ; get VBE mode
11458                              <1> vbe_gm_0:
11459 00003B17 66A1[1E120300]      <1>       mov    ax, [video_mode]
11460                              <1>              ; BX bits:
11461                              <1>              ; bit 0 to 8 = VESA VBE video mode
11462                              <1>              ; bit 9 to 13 = 0
11463                              <1>              ; bit 14 = last mode set LFB option
11464                              <1>              ;        1 - linear/flat frame buffer
11465                              <1>              ;        0 - windowed frame buffer
11466                              <1>              ; bit 15 = last mode set no_clear option
11467                              <1>              ;        0 - video memory cleared
11468                              <1>              ;        1 - video memory not cleared
11469                              <1>
11470                              <1> vbe_gm_return:
11471                              <1>       ;pop   edx ; ******
11472 00003B1D 0FB7D8              <1>       movzx  ebx, ax
11473                              <1> ;vbe_srs_retn:
11474 00003B20 31C0                <1>       xor    eax, eax ; 0
11475 00003B22 B04F                <1>       mov    al, 4Fh ; ax = 004Fh (successful)
11476 00003B24 C3                  <1>       retn
11477                              <1>
11478                              <1> vbe_gm_1:
11479                              <1>       ; legacy (old, standard) CGA/VGA bios video mode
11480 00003B25 8A25[87950100]      <1>       mov    ah, [noclearmem] ; 80h or 0
11481                              <1>              ; BX bits:
11482                              <1>              ; bit 0 to 7 = video mode
11483                              <1>              ; bit 8 to 13 = 0
11484                              <1>              ; bit 14 = 0 (not LFB mode) CGA/VGA
11485                              <1>              ; bit 15 = 1 if [noclearmem] = 80h
11486                              <1>              ;        0 if [noclearmem] = 0
11487 00003B2B EBF0                <1>       jmp    short vbe_gm_return
11488                              <1>
11489                              <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
11490                              <1> ; * ------------------------------------------------------
11491                              <1> ; * Function 04h - Save/Restore State
11492                              <1> ; * ------------------------------------------------------
11493                              <1> ; * Input:
11494                              <1> ; *          AX = 4F04h
11495                              <1> ; *              DL = 00h Return Save/Restore State buff size
11496                              <1> ; *                   01h Save State
11497                              <1> ; *                   02h Restore State
11498                              <1> ; *              CX = Requested states
11499                              <1> ; *                bit 0 - controller hardware state
11500                              <1> ; *                bit 1 - BIOS data state
```

```
11501                                  <1> ; *              bit 2 - DAC state
11502                                  <1> ; *              bit 3 - register state
11503                                  <1> ; *    (ES:BX) EBX = Pointer to buffer (if DL <> 00h)
11504                                  <1> ; * Output:
11505                                  <1> ; *         AX = VBE Return Status
11506                                  <1> ; *         BX = Number of 64-byte blocks
11507                                  <1> ; *              to hold the state buffer (if DL=00h)
11508                                  <1> ; *
11509                                  <1> ; *-------------------------------------------------------------
11510                                  <1> ; *
11511                                  <1>
11512                                  <1> vbe_biosfn_save_restore_state:
11513                                  <1>       ; 23/01/2021
11514                                  <1>       ; 16/01/2021
11515                                  <1>       ; 14/01/2021
11516                                  <1>       ; 13/01/2021
11517                                  <1>       ; 12/01/2021
11518                                  <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
11519                                  <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
11520                                  <1>       ;
11521                                  <1>       ; Input:
11522                                  <1>       ;      dl = sub function
11523                                  <1>       ;      cl = requested state
11524                                  <1>       ;      ebx = pointer to buffer (if dl<>00h)
11525                                  <1>       ; Output:
11526                                  <1>       ;      ax = 004Fh (successful)
11527                                  <1>       ;      ah > 0 -> error
11528                                  <1>       ;      bx = Number of 64-byte blocks
11529                                  <1>       ;           to hold the state buffer (if DL=00h)
11530                                  <1>
11531                                  <1>       ; Modified registers: eax, ebx, edi
11532                                  <1>
11533                                  <1>       ; 14/01/2021
11534 00003B2D 09DB                   <1>       or    ebx, ebx ; user's buffer address
11535 00003B2F 750A                   <1>       jnz   short _vbe_biosfn_save_restore_state
11536                                  <1>
11537 00003B31 20D2                   <1>       and   dl, dl
11538 00003B33 7406                   <1>       jz    short _vbe_biosfn_save_restore_state
11539                                  <1>
11540                                  <1>       ; function failed
11541                                  <1>       ;mov   eax, 0100h
11542                                  <1>       ;xor   eax, eax
11543                                  <1>       ;inc   ah  ; eax = 0100h
11544                                  <1>       ; 16/01/2021
11545 00003B35 B84F010000             <1>       mov   eax, 014Fh
11546 00003B3A C3                     <1>       retn
11547                                  <1>
11548                                  <1> _vbe_biosfn_save_restore_state:
11549                                  <1>       ; 23/01/2021
11550                                  <1>       ; 14/01/2021
11551                                  <1>       ; ebx = 0 if the caller is kernel ('sysvideo')
11552                                  <1>
11553                                  <1>       ; 13/01/2021
11554 00003B3B 57                     <1>       push  edi
11555 00003B3C 52                     <1>       push  edx
11556 00003B3D 51                     <1>       push  ecx
11557                                  <1>
11558                                  <1>       ; 23/01/2021
11559                                  <1>       ; 12/01/2021
11560 00003B3E 80FA02                 <1>       cmp   dl, 2
11561 00003B41 7757                   <1>       ja    short vbe_srs_7 ; 23/01/2021
11562                                  <1>                     ; invalid sub function
11563 00003B43 83F90F                 <1>       cmp   ecx, 0Fh
11564 00003B46 7752                   <1>       ja    short vbe_srs_7 ; invalid !
11565                                  <1>
11566 00003B48 20D2                   <1>       and   dl, dl
11567 00003B4A 7515                   <1>       jnz   short vbe_srs_4
11568                                  <1>
11569                                  <1>       ; DL = 0
11570                                  <1>       ; Return Save/Restore State buffer size
11571                                  <1>
11572                                  <1>       ;mov   ebx, ecx
11573                                  <1>       ;shl   bl, 1
11574                                  <1>       ;mov   bx, [ebx+vbestatebufsize]
11575 00003B4C E881000000             <1>       call  vbe_srs_gbs
11576                                  <1>
11577                                  <1> ;     ; 11/01/2021
11578                                  <1> ;     test  cl, 8
11579                                  <1> ;     jz    short vbe_srs_3
11580                                  <1> ;     ; vbe_biosfn_read_video_state_size();
11581                                  <1> ;     ; return 9 * 2;
11582                                  <1> ;     mov   bl, 18 ; register state size
11583                                  <1> ;vbe_srs_0:
11584                                  <1> ;     test  cl, 1
11585                                  <1> ;     jz    short vbe_srs_1
11586                                  <1> ;     ; size += 0x46;
11587                                  <1> ;     add   bl, 70 ; controller state size
11588                                  <1> ;vbe_srs_1:
11589                                  <1> ;     test  cl, 2
11590                                  <1> ;     jz    short vbe_srs_2
11591                                  <1> ;     ; size += (5 + 8 + 5) * 2 + 6;
11592                                  <1> ;     ;add  bl, 42 ; BIOS data state size ; Bochs/Plex86
11593                                  <1> ;     ; 12/01/2021
11594                                  <1> ;     add   bl, 40 ; TRDOS 386 v2 VBIOS data state size
11595                                  <1> ;vbe_srs_2:
11596                                  <1> ;     test  cl, 4
11597                                  <1> ;     jz    short vbe_srs_3
11598                                  <1> ;     ; size += 3 + 256 * 3 + 1;
11599                                  <1> ;     add   bx, 772 ; DAC state size
11600                                  <1>
11601                                  <1> vbe_srs_3:
11602 00003B51 6683C33F               <1>       add   bx, 63
11603 00003B55 66C1EB06               <1>       shr   bx, 6 ; / 64
11604                                  <1>
11605                                  <1> vbe_srs_retn:
```

```
11606 00003B59 31C0              <1>       xor    eax, eax ; 0
11607                            <1> vbe_srs_0:  ; 16/01/2021
11608 00003B5B B04F              <1>       mov    al, 4Fh ; ax = 004Fh (successful)
11609                            <1> ;vbe_srs_0:
11610                            <1>       ; 13/01/2021
11611 00003B5D 59                <1>       pop    ecx
11612 00003B5E 5A                <1>       pop    edx
11613 00003B5F 5F                <1>       pop    edi
11614                            <1>
11615 00003B60 C3                <1>       retn
11616                            <1>
11617                            <1>       ; 23/01/2021
11618                            <1> ;vbe_srs_10:
11619                            <1>       ;; 14/01/2021
11620                            <1>       ; return to 'sysvideo'
11621                            <1>       ;mov   ebx, ecx ; transfer count
11622                            <1>       ;      ; (byte count for saving current video state)
11623                            <1>       ;jmp   short vbe_srs_retn
11624                            <1>
11625                            <1> vbe_srs_4:
11626                            <1>       ; 23/01/2021
11627 00003B61 80E10F            <1>       and    cl, 0Fh ; 8, 4, 2, 1
11628 00003B64 7434              <1>       jz     short vbe_srs_7 ; cx = 0 -> invalid !
11629                            <1>
11630 00003B66 BF00760900        <1>       mov    edi, VBE3SAVERESTOREBLOCK
11631                            <1>
11632 00003B6B 80FA01            <1>       cmp    dl, 1
11633 00003B6E 7730              <1>       ja     short vbe_srs_8
11634                            <1>
11635                            <1>       ; save video state
11636                            <1>
11637 00003B70 F6C107            <1>       test   cl, 07h ; 4, 2, 1
11638 00003B73 740A              <1>       jz     short vbe_srs_5  ; vbe dispi regs state
11639                            <1>
11640 00003B75 E884000000        <1>       call   biosfn_save_video_state
11641                            <1>       ; edi = current position
11642                            <1>       ;      in VBE3SAVERESTOREBLOCK
11643                            <1>       ;      (VGA save_state offset)
11644                            <1>       ; modified regs: edi, eax, edx, ch
11645 00003B7A F6C108            <1>       test   cl, 8
11646 00003B7D 7405              <1>       jz     short vbe_srs_6
11647                            <1> vbe_srs_5:
11648 00003B7F E8AC010000        <1>       call   vbe_biosfn_save_video_state
11649                            <1>       ; edi = end position
11650                            <1>       ;      in VBE3SAVERESTOREBLOCK
11651                            <1>       ;      (VGA save_state offset)
11652                            <1>       ; modified regs: edi, eax, edx, ch
11653                            <1> vbe_srs_6:
11654                            <1>       ; 23/01/2021
11655 00003B84 21DB              <1>       and    ebx, ebx
11656 00003B86 74D1              <1>       jz     short vbe_srs_retn ; the caller is kernel
11657                            <1>
11658 00003B88 BE00760900        <1>       mov    esi, VBE3SAVERESTOREBLOCK
11659 00003B8D 29F7              <1>       sub    edi, esi
11660 00003B8F 89F9              <1>       mov    ecx, edi ; transfer count in bytes
11661                            <1>
11662                            <1>       ;; 14/01/2021
11663                            <1>       ;and   ebx, ebx
11664                            <1>       ;jz    short vbe_srs_10 ; the caller is kernel
11665                            <1>
11666 00003B91 89DF              <1>       mov    edi, ebx ; user's buffer address
11667 00003B93 E88CDE0000        <1>       call   transfer_to_user_buffer
11668 00003B98 73BF              <1>       jnc    short vbe_srs_retn
11669                            <1> vbe_srs_7:
11670                            <1>       ; // function failed
11671                            <1>       ;mov   eax, 0100h
11672 00003B9A 31C0              <1>       xor    eax, eax
11673 00003B9C FEC4              <1>       inc    ah  ; eax = 0100h
11674                            <1>       ; 16/01/2021
11675                            <1>       ; ax = 0014Fh
11676                            <1>       ;retn
11677                            <1>       ; 13/01/2021
11678 00003B9E EBBB              <1>       jmp    short vbe_srs_0
11679                            <1> vbe_srs_8:
11680                            <1>       ;cmp   dl, 2
11681                            <1>       ;jne   short vbe_srs_7
11682                            <1>       ;             ; invalid sub function
11683                            <1>
11684                            <1>       ; 14/01/2021
11685 00003BA0 09DB              <1>       or     ebx, ebx ; user's buffer address
11686                            <1>       ;jnz   short vbe_srs_11
11687                            <1>
11688                            <1>       ; the caller is kernel ('sysvideo')
11689                            <1>       ;jmp   short vbe_srs_12
11690                            <1>       ; 23/01/2021
11691 00003BA2 7414              <1>       jz     short vbe_srs_12 ; 'sysvideo' call
11692                            <1> vbe_srs_11:
11693 00003BA4 89DE              <1>       mov    esi, ebx ; user's buffer address
11694                            <1>       ; 23/01/2021
11695                            <1>       ;push  ebx
11696                            <1>
11697 00003BA6 E827000000        <1>       call   vbe_srs_gbs
11698                            <1>
11699                            <1>       ; restore video state
11700                            <1>
11701                            <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
11702 00003BAB 51                <1>       push   ecx
11703 00003BAC 89D9              <1>       mov    ecx, ebx ; transfer count in bytes
11704 00003BAE E8BBDE0000        <1>       call   transfer_from_user_buffer
11705 00003BB3 59                <1>       pop    ecx
11706                            <1>       ; 23/01/2021
11707                            <1>       ;pop   ebx
11708 00003BB4 89F3              <1>       mov    ebx, esi
11709 00003BB6 72E2              <1>       jc     short vbe_srs_7
11710                            <1>
```

```
11711                                  <1> vbe_srs_12:
11712                                  <1>       ;mov   esi, VBE3SAVERESTOREBLOCK
11713 00003BB8 89FE                    <1>       mov    esi, edi
11714                                  <1>
11715 00003BBA F6C107                  <1>       test   cl, 07h ; 4, 2, 1
11716 00003BBD 740C                    <1>       jz     short vbe_srs_9     ; vbe dispi regs state
11717                                  <1>
11718 00003BBF E8A8010000              <1>       call   biosfn_restore_video_state
11719 00003BC4 72D4                    <1>       jc     short vbe_srs_7 ; invalid buffer content !
11720                                  <1>       ; esi = current position
11721                                  <1>       ;      in VBE3SAVERESTOREBLOCK
11722                                  <1>       ;      (VGA save_state offset)
11723                                  <1>       ; modified regs: esi, eax, edx, ch
11724 00003BC6 F6C108                  <1>       test   cl, 8
11725                                  <1>       ;jz    short vbe_srs_10
11726                                  <1>       ; 23/01/2020
11727 00003BC9 EB8E                    <1>       jmp    short vbe_srs_retn
11728                                  <1> vbe_srs_9:
11729 00003BCB E8F8020000              <1>       call   vbe_biosfn_restore_video_state
11730                                  <1>
11731                                  <1>       ; modified regs: esi, eax, edx, ch
11732                                  <1>
11733 00003BD0 EB87                    <1>       jmp    short vbe_srs_retn
11734                                  <1>
11735                                  <1> ;vbe_srs_10:
11736                                  <1> ;     ; successful
11737                                  <1> ;     xor    eax, eax ; 0
11738                                  <1> ;     mov    al, 4Fh ; ax = 004Fh (successful)
11739                                  <1> ;     retn
11740                                  <1>
11741                                  <1> vbe_srs_gbs:
11742                                  <1>       ; return buffer size according to flags
11743 00003BD2 89CB                    <1>       mov    ebx, ecx ; options/flags
11744 00003BD4 D0E3                    <1>       shl    bl, 1
11745 00003BD6 668B9B[DE3B0000]        <1>       mov    bx, [ebx+vbestatebufsize]
11746 00003BDD C3                      <1>       retn
11747                                  <1>
11748                                  <1> vbestatebufsize:
11749                                  <1>       ; ---------------------------------------
11750                                  <1>       ; CL = 0  1   2    3    4    5    6    7
11751                                  <1>       ; ---------------------------------------
11752 00003BDE 0000460028006E0004-     <1>       dw     0, 70, 40, 110, 772, 842, 812, 882
11752 00003BE7 034A032C037203          <1>
11753                                  <1>       ; ---------------------------------------
11754                                  <1>       ; CL = 8   9   10  11   12   13   14   15
11755                                  <1>       ; ---------------------------------------
11756 00003BEE 120058003A00800016-     <1>       dw     18, 88, 58, 128, 790, 860, 830, 900
11756 00003BF7 035C033E038403          <1>
11757                                  <1>
11758                                  <1> ; 11/01/2021
11759                                  <1> VGAREG_ACTL_ADDRESS       equ 3C0h
11760                                  <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
11761                                  <1> VGAREG_ACTL_READ_DATA     equ 3C1h
11762                                  <1>
11763                                  <1> VGAREG_INPUT_STATUS       equ 3C2h
11764                                  <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
11765                                  <1> VGAREG_VIDEO_ENABLE       equ 3C3h
11766                                  <1> VGAREG_SEQU_ADDRESS       equ 3C4h
11767                                  <1> VGAREG_SEQU_DATA   equ 3C5h
11768                                  <1>
11769                                  <1> VGAREG_PEL_MASK           equ 3C6h
11770                                  <1> VGAREG_DAC_STATE   equ 3C7h
11771                                  <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
11772                                  <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
11773                                  <1> VGAREG_DAC_DATA           equ 3C9h
11774                                  <1>
11775                                  <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
11776                                  <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
11777                                  <1>
11778                                  <1> VGAREG_GRDC_ADDRESS       equ 3CEh
11779                                  <1> VGAREG_GRDC_DATA   equ 3CFh
11780                                  <1>
11781                                  <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
11782                                  <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
11783                                  <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
11784                                  <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
11785                                  <1>
11786                                  <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
11787                                  <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
11788                                  <1> VGAREG_ACTL_RESET  equ 3DAh
11789                                  <1>
11790                                  <1> ;VGAREG_MDA_MODECTL       equ 3B8h
11791                                  <1> VGAREG_CGA_MODECTL equ 3D8h
11792                                  <1> VGAREG_CGA_PALETTE equ 3D9h
11793                                  <1>
11794                                  <1> biosfn_save_video_state:
11795                                  <1>       ; 22/01/2021
11796                                  <1>       ; 12/01/2021
11797                                  <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
11798                                  <1>       ; (vgabios.c)
11799                                  <1>
11800                                  <1>       ; modified registers: eax, edx, edi, ch
11801                                  <1>
11802                                  <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
11803                                  <1>
11804                                  <1>       ; input: edi = state buffer address
11805                                  <1>
11806 00003BFE F6C101                  <1>       test   cl, 1
11807 00003C01 0F8485000000            <1>       jz     bfn_svs_4
11808                                  <1>
11809 00003C07 66BAC403                <1>       mov    dx, VGAREG_SEQU_ADDRESS ; 3C7h
11810 00003C0B EC                      <1>       in     al, dx
11811 00003C0C AA                      <1>       stosb
11812                                  <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11813 00003C0D B2D4                    <1>       mov    dl, 0D4h
```

```
11814 00003C0F EC                <1>       in     al, dx
11815 00003C10 AA                <1>       stosb
11816                            <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
11817 00003C11 B2CE              <1>         mov  dl, 0CEh
11818 00003C13 EC                <1>       in     al, dx
11819 00003C14 AA                <1>       stosb
11820                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
11821 00003C15 B2DA              <1>       mov    dl, 0DAh
11822 00003C17 EC                <1>       in     al, dx
11823                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
11824 00003C18 B2C0              <1>       mov    dl, 0C0h
11825 00003C1A EC                <1>       in     al, dx
11826 00003C1B AA                <1>       stosb
11827 00003C1C 88C4              <1>       mov    ah, al ; ar_index
11828                            <1>       ;mov   dx, VGAREG_READ_FEATURE_CTL ; 3CAh
11829 00003C1E B2CA              <1>       mov    dl, 0CAh
11830 00003C20 EC                <1>       in     al, dx
11831 00003C21 AA                <1>       stosb
11832                            <1>       ; (5 bytes are writen above)
11833                            <1>
11834                            <1>       ; for(i=1;i<=4;i++){
11835 00003C22 B001              <1>       mov    al, 1
11836                            <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
11837                            <1>       ;mov   dl, 0C4h
11838 00003C24 B504              <1>       mov    ch, 4
11839                            <1> bfn_svs_0:
11840                            <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
11841                            <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
11842 00003C26 B2C4              <1>       mov    dl, 0C4h
11843 00003C28 EE                <1>       out    dx, al
11844                            <1>       ;mov   dx, VGAREG_SEQU_DATA  ; 3C5h
11845 00003C29 FEC2              <1>       inc    dl  ; dx = 3C5h
11846                            <1>       ; inb(VGAREG_SEQU_DATA)
11847 00003C2B 50                <1>       push   eax
11848 00003C2C EC                <1>       in     al, dx
11849 00003C2D AA                <1>       stosb  ; (4 bytes in loop)
11850 00003C2E 58                <1>       pop    eax
11851                            <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
11852                            <1>       ;dec   dl
11853 00003C2F FEC0              <1>       inc    al  ; i++
11854 00003C31 FECD              <1>       dec    ch
11855 00003C33 75F1              <1>       jnz    short bfn_svs_0
11856                            <1>
11857                            <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
11858 00003C35 28C0              <1>       sub    al, al ; 0
11859 00003C37 EE                <1>       out    dx, al
11860                            <1>       ; inb(VGAREG_SEQU_DATA)
11861                            <1>       ;mov   dx, VGAREG_SEQU_DATA ; 3C5h
11862 00003C38 FEC2              <1>       inc    dl  ; dx = 3C5h
11863 00003C3A EC                <1>       in     al, dx
11864 00003C3B AA                <1>       stosb  ; (+1 byte)
11865                            <1>
11866                            <1>         ; for(i=0;i<=0x18;i++) {
11867 00003C3C 28C0              <1>       sub    al, al ; 0
11868                            <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11869                            <1>       ;mov   dl, 0D4h
11870 00003C3E B519              <1>       mov    ch, 25
11871                            <1> bfn_svs_1:
11872                            <1>         ; outb(crtc_addr,i);
11873                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11874 00003C40 B2D4              <1>       mov    dl, 0D4h
11875 00003C42 EE                <1>       out    dx, al
11876                            <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
11877 00003C43 FEC2              <1>       inc    dl  ; dx = 3D5h
11878                            <1>       ; inb(crtc_addr+1)
11879 00003C45 50                <1>       push   eax
11880 00003C46 EC                <1>       in     al, dx
11881 00003C47 AA                <1>       stosb  ; (25 bytes in loop)
11882 00003C48 58                <1>       pop    eax
11883                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11884                            <1>       ;dec   dl
11885 00003C49 FEC0              <1>       inc    al  ; i++
11886 00003C4B FECD              <1>       dec    ch
11887 00003C4D 75F1              <1>       jnz    short bfn_svs_1
11888                            <1>
11889 00003C4F 80E420            <1>       and    ah, 20h  ; (ar_index & 0x20)
11890                            <1>         ; for(i=0;i<=0x13;i++) {
11891 00003C52 28C0              <1>       sub    al, al ; 0
11892 00003C54 B514              <1>       mov    ch, 20
11893                            <1> bfn_svs_2:
11894                            <1>       ; inb(VGAREG_ACTL_RESET);
11895                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
11896 00003C56 B2DA              <1>       mov    dl, 0DAh
11897 00003C58 50                <1>       push   eax
11898 00003C59 EC                <1>       in     al, dx
11899 00003C5A 8A0424            <1>       mov    al, [esp]
11900                            <1>       ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
11901 00003C5D 08E0              <1>       or     al, ah
11902                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
11903 00003C5F B2C0              <1>       mov    dl, 0C0h
11904 00003C61 EE                <1>       out    dx, al
11905                            <1>       ;mov   dx, VGAREG_ACTL_READ_DATA ; 3C1h
11906                            <1>       ;mov   dl, 0C1h
11907 00003C62 FEC2              <1>       inc    dl
11908 00003C64 EC                <1>       in     al, dx
11909 00003C65 AA                <1>       stosb  ; (20 bytes in loop)
11910 00003C66 58                <1>       pop    eax
11911 00003C67 FEC0              <1>       inc    al  ; i++
11912 00003C69 FECD              <1>       dec    ch
11913 00003C6B 75E9              <1>       jnz    short bfn_svs_2
11914                            <1>
11915                            <1>       ; inb(VGAREG_ACTL_RESET);
11916                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
11917 00003C6D B2DA              <1>       mov    dl, 0DAh
11918 00003C6F EC                <1>       in     al, dx
```

```
11919                                  <1>
11920                                  <1>         ; for(i=0;i<=8;i++) {
11921 00003C70 28C0                     <1>       sub   al, al ; 0
11922                                  <1>       ;;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
11923                                  <1>       ;mov   dl, 0CEh
11924 00003C72 B509                     <1>       mov   ch, 9
11925                                  <1> bfn_svs_3:
11926                                  <1>       ; outb(VGAREG_GRDC_ADDRESS,i)
11927                                  <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
11928 00003C74 B2CE                     <1>       mov   dl, 0CEh
11929 00003C76 EE                       <1>       out   dx, al
11930                                  <1>       ; inb(VGAREG_ACTL_READ_DATA)
11931 00003C77 50                       <1>       push  eax
11932                                  <1>       ;mov   dx, VGAREG_GRDC_DATA ; 3CFh
11933                                  <1>       ;mov   dl, 0CFh
11934 00003C78 FEC2                     <1>       inc   dl
11935 00003C7A EC                       <1>       in    al, dx
11936 00003C7B AA                       <1>       stosb ; (9 bytes in loop)
11937 00003C7C 58                       <1>       pop   eax
11938                                  <1>       ;dec   dl
11939 00003C7D FEC0                     <1>       inc   al  ; i++
11940 00003C7F FECD                     <1>       dec   ch
11941 00003C81 75F1                     <1>       jnz   short bfn_svs_3
11942                                  <1>
11943                                  <1>       ; write_word(ES, BX, crtc_addr); BX+= 2;
11944                                  <1>       ; (offset 64)
11945 00003C83 66B8D403                 <1>       mov   ax, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
11946 00003C87 66AB                     <1>       stosw ; (2 bytes (1 word))
11947                                  <1>
11948                                  <1>         ; /* XXX: read plane latches */
11949 00003C89 31C0                     <1>       xor   eax, eax  ; 0
11950 00003C8B AB                       <1>             stosd ; (4 bytes)
11951                                  <1>
11952                                  <1>       ; (total 70 bytes are written above as controller hardware state)
11953                                  <1>
11954                                  <1> bfn_svs_4:
11955                                  <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
11956 00003C8C F6C102                   <1>       test  cl, 2
11957 00003C8F 7476                     <1>       jz    short bfn_svs_6
11958                                  <1>
11959                                  <1>       ; VIDEO BIOS DATA
11960                                  <1>       ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
11961                                  <1>       ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
11962                                  <1>
11963                                  <1>       ; if (CX & 2) {
11964                                  <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE)); BX++;
11965                                  <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)); BX += 2;
11966                                  <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE)); BX += 2;
11967                                  <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS)); BX += 2;
11968                                  <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS)); BX++;
11969                                  <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT)); BX += 2;
11970                                  <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL)); BX++;
11971                                  <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES)); BX++;
11972                                  <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL)); BX++;
11973                                  <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE)); BX += 2;
11974                                  <1>         ;for(i=0;i<8;i++) {
11975                                  <1>         ;   write_word(ES, BX, read_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i));
11976                                  <1>         ;   BX += 2;
11977                                  <1>         ;}
11978                                  <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START)); BX += 2;
11979                                  <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE)); BX++;
11980                                  <1>         ;/* current font */
11981                                  <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4)); BX += 2;
11982                                  <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4 + 2)); BX += 2;
11983                                  <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4)); BX += 2;
11984                                  <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4 + 2)); BX += 2;
11985                                  <1>
11986                                  <1>       ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
11987                                  <1>       ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
11988                                  <1>
11989 00003C91 66B8D403                 <1>       mov   ax, 3D4h ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
11990 00003C95 66AB                     <1>       stosw
11991 00003C97 A0[9A6E0000]             <1>       mov   al, [CRT_MODE] ; Current video mode (0FFh for VESA VBE modes)
11992 00003C9C AA                       <1>       stosb
11993 00003C9D A0[9B6E0000]             <1>       mov   al, [CRT_MODE_SET] ; 29h for mode 03h ; TRDOS 386 feature only !
11994 00003CA2 AA                       <1>       stosb
11995 00003CA3 66A1[1E120300]           <1>       mov   ax, [video_mode] ; Current VESA VBE (SVGA, extended VGA) mode
11996 00003CA9 66AB                     <1>       stosw               ; (valid if [CRT_MODE] = 0FFh)
11997 00003CAB 66A1[88950100]           <1>       mov   ax, [CRT_LEN] ; page size (in bytes)
11998 00003CB1 66AB                     <1>       stosw
11999 00003CB3 66A1[0C890100]           <1>       mov   ax, [CRT_START] ; video page start offset
12000 00003CB9 66AB                     <1>       stosw
12001 00003CBB A0[9C6E0000]             <1>       mov   al, [CRT_COLS] ; nbcols, characters per row
12002 00003CC0 AA                       <1>       stosb
12003 00003CC1 A0[A26E0000]             <1>       mov   al, [VGA_ROWS] ; nbrows, (character) rows per page (not rows-1)
12004 00003CC6 AA                       <1>       stosb
12005 00003CC7 A0[9E6E0000]             <1>       mov   al, [CHAR_HEIGHT] ; character font height (8 or 16 or 14)
12006 00003CCC AA                       <1>       stosb
12007 00003CCD A0[9F6E0000]             <1>       mov   al, [VGA_VIDEO_CTL] ; ROM BIOS DATA AREA Offset 87h
12008 00003CD2 AA                       <1>       stosb
12009 00003CD3 A0[A06E0000]             <1>       mov   al, [VGA_SWITCHES] ; feature bit switches
12010 00003CD8 AA                       <1>       stosb
12011 00003CD9 A0[A16E0000]             <1>       mov   al, [VGA_MODESET_CTL] ; basic mode set options
12012 00003CDE AA                       <1>       stosb
12013                                  <1>       ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
12014                                  <1>       ; (bochs/plex86 does not use and return those)
12015 00003CDF A0[9D6E0000]             <1>       mov   al, [CRT_PALETTE] ; current color palette ; TRDOS 386 feature only !
12016 00003CE4 AA                       <1>       stosb
12017 00003CE5 A0[1E890100]             <1>       mov   al, [ACTIVE_PAGE] ; current video page
12018 00003CEA AA                       <1>       stosb
12019 00003CEB 66A1[B36E0000]           <1>       mov   ax, [CURSOR_MODE] ; cursor type
12020 00003CF1 66AB                     <1>       stosw
12021                                  <1>       ;mov   eax, [CURSOR_POSN] ; cursor position for video page 0 and 1
12022                                  <1>       ;stosd
12023                                  <1>       ;mov   eax, [CURSOR_POSN+4] ; cursor position for video page 2 and 3
```

```
12024                                  <1>       ;stosd
12025                                  <1>       ;mov   eax, [CURSOR_POSN+8] ; cursor position for video page 4 and 5
12026                                  <1>       ;stosd
12027                                  <1>       ;mov   eax, [CURSOR_POSN+12] ; cursor position for video page 6 and 7
12028                                  <1>       ;stosd
12029 00003CF3 56                      <1>       push   esi
12030 00003CF4 B504                    <1>       mov    ch, 4
12031 00003CF6 BE[0E890100]            <1>       mov    esi, CURSOR_POSN
12032                                  <1> bfn_svs_5:
12033 00003CFB A5                      <1>       movsd
12034 00003CFC FECD                    <1>       dec    ch
12035 00003CFE 75FB                    <1>       jnz    short bfn_svs_5
12036 00003D00 5E                      <1>       pop    esi
12037                                  <1>       ; (font addr) protected mode address in kernel's/system memory space
12038                                  <1>       ; (not accessable/meaningful address value by user)
12039 00003D01 A1[9A950100]            <1>       mov    eax, [VGA_INT43H] ; VGA current (default) font address
12040 00003D06 AB                      <1>       stosd
12041                                  <1>
12042                                  <1>       ; (total 40 bytes are written above as BIOS data state)
12043                                  <1>
12044                                  <1> bfn_svs_6:
12045                                  <1>       ; 12/01/2021
12046 00003D07 F6C104                  <1>       test   cl, 4
12047 00003D0A 7423                    <1>       jz     short bfn_svs_8
12048                                  <1>
12049                                  <1>       ;/* XXX: check this */
12050                                  <1>              ; /* read/write mode dac */
12051                                  <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_STATE)); BX++;
12052                                  <1>       ;      ; /* pix address */
12053                                  <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_WRITE_ADDRESS)); BX++;
12054                                  <1>         ;write_byte(ES, BX, inb(VGAREG_PEL_MASK)); BX++;
12055                                  <1>         ;// Set the whole dac always, from 0
12056                                  <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
12057                                  <1>         ;for(i=0;i<256*3;i++) {
12058                                  <1>         ;   write_byte(ES, BX, inb(VGAREG_DAC_DATA)); BX++;
12059                                  <1>         ;}
12060                                  <1>         ;write_byte(ES, BX, 0); BX++; /* color select register */
12061                                  <1>
12062                                  <1>       ; /* read/write mode dac */
12063 00003D0C 66BAC703                <1>       mov    dx, 3C7h ; VGAREG_DAC_STATE
12064 00003D10 EC                      <1>       in     al, dx
12065 00003D11 AA                      <1>       stosb
12066                                  <1>       ; /* pix address */
12067                                  <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
12068                                  <1>       ;mov   dl, 0C8h
12069 00003D12 FEC2                    <1>       inc    dl
12070 00003D14 EC                      <1>       in     al, dx
12071 00003D15 AA                      <1>       stosb
12072                                  <1>       ;mov   dx, VGAREG_PEL_MASK  ; 3C6h
12073 00003D16 B2C6                    <1>       mov    dl, 0C6h
12074 00003D18 EC                      <1>       in     al, dx
12075 00003D19 AA                      <1>       stosb
12076                                  <1>       ;// Set the whole dac always, from 0
12077 00003D1A 30C0                    <1>       xor    al, al ; 0
12078                                  <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
12079 00003D1C B2C8                    <1>       mov    dl, 0C8h
12080 00003D1E EE                      <1>       out    dx, al
12081                                  <1>
12082 00003D1F 51                      <1>       push   ecx ; 22/01/2021
12083                                  <1>       ;for(i=0;i<256*3;i++) {
12084 00003D20 B900030000              <1>       mov    ecx, 256*3 ; 768 bytes
12085                                  <1>       ;mov   dx, VGAREG_DAC_DATA ; 3C9h
12086                                  <1>       ;mov   dl, 0C9h
12087 00003D25 FEC2                    <1>       inc    dl ; dx = 3C9h
12088                                  <1> bfn_svs_7:
12089 00003D27 EC                      <1>       in     al, dx
12090 00003D28 AA                      <1>       stosb
12091 00003D29 E2FC                    <1>       loop   bfn_svs_7
12092 00003D2B 59                      <1>       pop    ecx ; 22/01/2021
12093                                  <1>
12094                                  <1>       ; /* color select register */
12095 00003D2C 28C0                    <1>       sub    al, al ; 0
12096 00003D2E AA                      <1>       stosb
12097                                  <1>
12098                                  <1>       ; (total 772 bytes are written above as DAC state)
12099                                  <1> bfn_svs_8:
12100 00003D2F C3                      <1>       retn
12101                                  <1>
12102                                  <1> vbe_biosfn_save_video_state:
12103                                  <1>       ; 23/01/2021
12104                                  <1>       ; 13/01/2021
12105                                  <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
12106                                  <1>       ; (vbe.c)
12107                                  <1>
12108                                  <1>       ; modified registers: eax, edx, edi, ch
12109                                  <1>
12110                                  <1>       ; input: edi = state buffer address
12111                                  <1>       ; output:
12112                                  <1>       ;       VBE DISPI register contents will be saved
12113                                  <1>       ;       (18 bytes, 9 words)
12114                                  <1>
12115                                  <1>       ; outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
12116                                  <1>       ; enable = inw(VBE_DISPI_IOPORT_DATA);
12117                                  <1>       ; write_word(ES, BX, enable);
12118                                  <1>       ; BX += 2;
12119                                  <1>       ; if (!(enable & VBE_DISPI_ENABLED))
12120                                  <1>       ;     return;
12121                                  <1>       ; for(i = VBE_DISPI_INDEX_XRES;
12122                                  <1>       ;             i <= VBE_DISPI_INDEX_Y_OFFSET; i++) {
12123                                  <1>       ;    if (i != VBE_DISPI_INDEX_ENABLE) {
12124                                  <1>       ;        outw(VBE_DISPI_IOPORT_INDEX, i);
12125                                  <1>       ;        write_word(ES, BX, inw(VBE_DISPI_IOPORT_DATA));
12126                                  <1>         ;    BX += 2;
12127                                  <1>       ;        }
12128                                  <1>       ; }
```

```
12129                                  <1>
12130 00003D30 66BACE01                <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12131 00003D34 B804000000              <1>       mov    eax, 04h  ; VBE_DISPI_INDEX_ENABLE
12132 00003D39 66EF                    <1>       out    dx, ax
12133                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12134 00003D3B FEC2                    <1>       inc    dl
12135 00003D3D 66ED                    <1>       in     ax, dx ; enable (status)
12136 00003D3F 66AB                    <1>       stosw
12137 00003D41 6683E001                <1>       and    ax, 1 ; VBE_DISPI_ENABLED
12138 00003D45 7505                    <1>       jnz    short vbe_bfn_svs_0
12139                                  <1>       ; 23/01/2021
12140                                  <1>       ; ax = 0
12141                                  <1>       ; VBE_DISPI_DISABLED
12142                                  <1>       ; 13/01/2021
12143                                  <1>       ; clear remain 8 bytes
12144                                  <1>       ;xor   eax, eax
12145 00003D47 AB                     <1>       stosd  ; 2
12146 00003D48 AB                     <1>       stosd  ; 2
12147 00003D49 AB                     <1>       stosd  ; 2
12148 00003D4A AB                     <1>       stosd  ; 2
12149 00003D4B C3                     <1>       retn
12150                                  <1> vbe_bfn_svs_0:
12151                                  <1>       ; VBE_DISPI_ENABLED
12152                                  <1>
12153                                  <1>       ;sub   eax, eax
12154 00003D4C 28C0                    <1>       sub    al, al ; eax = 0
12155                                  <1>
12156                                  <1>       ; from VBE_DISPI_INDEX_XRES
12157                                  <1>       ;   to VBE_DISPI_INDEX_BPP
12158                                  <1>
12159 00003D4E B503                    <1>       mov    ch, 3
12160                                  <1>       ; al = 0 ; VBE_DISPI_INDEX_XRES - 1
12161                                  <1>
12162 00003D50 E804000000              <1>       call   vbe_bfn_svs_1
12163                                  <1>
12164                                  <1>       ; from VBE_DISPI_INDEX_BANK
12165                                  <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
12166                                  <1>
12167 00003D55 FEC0                    <1>       inc    al
12168                                  <1>       ; al = 4 ; VBE_DISPI_INDEX_BANK - 1
12169                                  <1>
12170 00003D57 B505                    <1>       mov    ch, 5
12171                                  <1> vbe_bfn_svs_1:
12172 00003D59 FEC0                    <1>       inc    al ; from VBE_DISPI_INDEX_XRES
12173                                  <1>                  ;   to VBE_DISPI_INDEX_BPP
12174                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12175 00003D5B FECA                    <1>       dec    dl ; 1CEh
12176 00003D5D 66EF                    <1>       out    dx, ax
12177 00003D5F 50                      <1>       push   eax
12178                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12179 00003D60 FEC2                    <1>       inc    dl ; 1CFh
12180 00003D62 66ED                    <1>       in     ax, dx
12181 00003D64 66AB                    <1>       stosw
12182 00003D66 58                      <1>       pop    eax
12183 00003D67 FECD                    <1>       dec    ch
12184 00003D69 75EE                    <1>       jnz    short vbe_bfn_svs_1
12185 00003D6B C3                      <1>       retn
12186                                  <1>
12187                                  <1> ; 11/01/2021
12188                                  <1> VGAREG_ACTL_ADDRESS       equ 3C0h
12189                                  <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
12190                                  <1> VGAREG_ACTL_READ_DATA     equ 3C1h
12191                                  <1>
12192                                  <1> VGAREG_INPUT_STATUS       equ 3C2h
12193                                  <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
12194                                  <1> VGAREG_VIDEO_ENABLE       equ 3C3h
12195                                  <1> VGAREG_SEQU_ADDRESS       equ 3C4h
12196                                  <1> VGAREG_SEQU_DATA   equ 3C5h
12197                                  <1>
12198                                  <1> VGAREG_PEL_MASK           equ 3C6h
12199                                  <1> VGAREG_DAC_STATE   equ 3C7h
12200                                  <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
12201                                  <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
12202                                  <1> VGAREG_DAC_DATA           equ 3C9h
12203                                  <1>
12204                                  <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
12205                                  <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
12206                                  <1>
12207                                  <1> VGAREG_GRDC_ADDRESS       equ 3CEh
12208                                  <1> VGAREG_GRDC_DATA   equ 3CFh
12209                                  <1>
12210                                  <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
12211                                  <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
12212                                  <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
12213                                  <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
12214                                  <1>
12215                                  <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
12216                                  <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
12217                                  <1> VGAREG_ACTL_RESET  equ 3DAh
12218                                  <1>
12219                                  <1> ;VGAREG_MDA_MODECTL       equ 3B8h
12220                                  <1> VGAREG_CGA_MODECTL equ 3D8h
12221                                  <1> VGAREG_CGA_PALETTE equ 3D9h
12222                                  <1>
12223                                  <1> biosfn_restore_video_state:
12224                                  <1>       ; 22/01/2021
12225                                  <1>       ; 13/01/2021
12226                                  <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
12227                                  <1>       ; (vgabios.c)
12228                                  <1>
12229                                  <1>       ; modified registers: eax, edx, esi, edi, ch
12230                                  <1>
12231                                  <1>       ;mov   esi, VBE3SAVERESTOREBLOCK
12232                                  <1>
12233                                  <1>       ; input: esi = state buffer address
```

```
12234                                  <1>
12235 00003D6C F6C101                  <1>       test   cl, 1
12236 00003D6F 0F84A9000000            <1>       jz     bfn_rvs_6
12237                                  <1>
12238 00003D75 66817E40D403            <1>       cmp    word [esi+64], 3D4h ; must be 3D4h
12239 00003D7B 7402                    <1>       je     short bfn_rvs_0
12240                                  <1>                     ; it is seen as valid buffer
12241 00003D7D F9                      <1>       stc
12242 00003D7E C3                      <1>       retn
12243                                  <1>
12244                                  <1> bfn_rvs_0:
12245 00003D7F 89F7                    <1>       mov    edi, esi ; addr1
12246 00003D81 83C605                  <1>       add    esi, 5 ; skip 1st 5 bytes for now
12247                                  <1>
12248                                  <1>       ; // Reset Attribute Ctl flip-flop
12249                                  <1>         ; inb(VGAREG_ACTL_RESET);
12250 00003D84 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
12251 00003D88 EC                      <1>       in     al, dx
12252                                  <1>
12253                                  <1>       ; for(i=1;i<=4;i++){
12254 00003D89 B001                    <1>       mov    al, 1
12255                                  <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
12256                                  <1>       ;mov   dl, 0C4h
12257 00003D8B B504                    <1>       mov    ch, 4
12258                                  <1> bfn_rvs_1:
12259                                  <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
12260                                  <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
12261 00003D8D B2C4                    <1>       mov    dl, 0C4h
12262 00003D8F EE                      <1>       out    dx, al
12263                                  <1>       ;mov   dx, VGAREG_SEQU_DATA  ; 3C5h
12264 00003D90 FEC2                    <1>       inc    dl  ; dx = 3C5h
12265                                  <1>       ; outb(VGAREG_SEQU_DATA)
12266 00003D92 50                      <1>       push   eax
12267 00003D93 AC                      <1>       lodsb  ; (4 bytes in loop)
12268 00003D94 EE                      <1>       out    dx, al
12269 00003D95 58                      <1>       pop    eax
12270                                  <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
12271                                  <1>       ;dec   dl
12272 00003D96 FEC0                    <1>       inc    al  ; i++
12273 00003D98 FECD                    <1>       dec    ch
12274 00003D9A 75F1                    <1>       jnz    short bfn_rvs_1
12275                                  <1>
12276                                  <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
12277 00003D9C 28C0                    <1>       sub    al, al ; 0
12278 00003D9E EE                      <1>       out    dx, al
12279                                  <1>       ; outb(VGAREG_SEQU_DATA)
12280                                  <1>       ;mov   dx, VGAREG_SEQU_DATA ; 3C5h
12281 00003D9F FEC2                    <1>       inc    dl  ; dx = 3C5h
12282 00003DA1 AC                      <1>       lodsb  ; (+1 byte)
12283 00003DA2 EE                      <1>       out    dx, al
12284                                  <1>
12285                                  <1>       ; // Disable CRTC write protection
12286                                  <1>       ; outw(crtc_addr,0x0011);
12287                                  <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12288 00003DA3 B2D4                    <1>       mov    dl, 0D4h
12289 00003DA5 66B81100                <1>       mov    ax, 11h
12290 00003DA9 66EF                    <1>       out    dx, ax
12291                                  <1>
12292                                  <1>       ; // Set CRTC regs
12293                                  <1>
12294                                  <1>       ; for(i=0;i<=0x18;i++) {
12295                                  <1>         ;   if (i != 0x11) {
12296 00003DAB 28C0                    <1>       sub    al, al ; 0
12297                                  <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12298                                  <1>       ;mov   dl, 0D4h
12299 00003DAD B519                    <1>       mov    ch, 25
12300                                  <1> bfn_rvs_2:
12301                                  <1>         ; outb(crtc_addr,i);
12302                                  <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12303 00003DAF B2D4                    <1>       mov    dl, 0D4h
12304 00003DB1 EE                      <1>       out    dx, al
12305                                  <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
12306 00003DB2 FEC2                    <1>       inc    dl  ; dx = 3D5h
12307                                  <1>       ; inb(crtc_addr+1)
12308 00003DB4 50                      <1>       push   eax
12309 00003DB5 AC                      <1>       lodsb  ; (25 bytes in loop)
12310 00003DB6 EE                      <1>       out    dx, al
12311 00003DB7 58                      <1>       pop    eax
12312                                  <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12313                                  <1>       ;dec   dl
12314 00003DB8 FEC0                    <1>       inc    al  ; i++
12315 00003DBA 3C11                    <1>       cmp    al, 17 ; 11h
12316 00003DBC 7505                    <1>       jne    short bfn_rvs_3
12317 00003DBE AC                      <1>       lodsb
12318 00003DBF 88C4                    <1>       mov    ah, al ; *
12319 00003DC1 B012                    <1>       mov    al, 18
12320                                  <1> bfn_rvs_3:
12321 00003DC3 FECD                    <1>       dec    ch
12322 00003DC5 75E8                    <1>       jnz    short bfn_rvs_2
12323                                  <1>
12324                                  <1>       ; // select crtc base address
12325                                  <1>         ; v = inb(VGAREG_READ_MISC_OUTPUT) & ~0x01;
12326                                  <1>       ;if (crtc_addr = 0x3d4)
12327                                  <1>       ;   v |= 0x01;
12328                                  <1>       ; outb(VGAREG_WRITE_MISC_OUTPUT, v);
12329                                  <1>
12330                                  <1>       ;;mov  dx, VGAREG_READ_MISC_OUTPUT ; 3CCh
12331                                  <1>       ;mov   dl, 0CCh
12332                                  <1>       ;in    al, dl
12333                                  <1>       ;and   al, 1
12334                                  <1>       ;;mov  dx, VGAREG_WRITE_MISC_OUTPUT ; 3C2h
12335                                  <1>       ;mov   dl, 0C2h
12336                                  <1>       ;or    al, 1
12337                                  <1>       ;out   dx, al
12338                                  <1>
```

```
12339                           <1>       ; // enable write protection if needed
12340                           <1>       ;outb(crtc_addr, 0x11);
12341                           <1>         ;outb(crtc_addr+1, read_byte(ES, BX - 0x18 + 0x11));
12342                           <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12343 00003DC7 B2D4            <1>       mov    dl, 0D4h
12344 00003DC9 B011            <1>       mov    al, 11h
12345 00003DCB EE              <1>       out    dx, al
12346 00003DCC 88E0            <1>       mov    al, ah ; *
12347 00003DCE FEC2            <1>       inc    dl ; dx = 3D5h
12348 00003DD0 EE              <1>       out    dx, al
12349                           <1>
12350                           <1>       ; // Set Attribute Ctl
12351 00003DD1 8A6703          <1>       mov    ah, [edi+3] ; addr1+3, ah = ar_index
12352 00003DD4 80E420          <1>       and    ah, 20h  ; (ar_index & 0x20)
12353                           <1>
12354                           <1>         ; inb(VGAREG_ACTL_RESET);
12355                           <1>       ;mov   dx, 3DAh ; VGAREG_ACTL_RESET
12356 00003DD7 B2DA            <1>       mov    dl, 0DAh
12357 00003DD9 EC              <1>       in     al, dx
12358                           <1>
12359                           <1>         ; for(i=0;i<=0x13;i++) {
12360 00003DDA 28C0            <1>       sub    al, al ; 0
12361 00003DDC B514            <1>       mov    ch, 20
12362                           <1> bfn_rvs_4:
12363                           <1>       ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
12364 00003DDE 50              <1>       push   eax
12365 00003DDF 08E0            <1>       or     al, ah
12366                           <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
12367 00003DE1 B2C0            <1>       mov    dl, 0C0h
12368 00003DE3 EE              <1>       out    dx, al
12369                           <1>       ;mov   dx, VGAREG_ACTL_WRITE_DATA ; 3C0h
12370                           <1>       ;mov   dl, 0C0h
12371 00003DE4 AC              <1>       lodsb  ; (20 bytes in loop)
12372 00003DE5 EE              <1>       out    dx, al
12373 00003DE6 58              <1>       pop    eax
12374 00003DE7 FEC0            <1>       inc    al  ; i++
12375 00003DE9 FECD            <1>       dec    ch
12376 00003DEB 75F1            <1>       jnz    short bfn_rvs_4
12377                           <1>
12378                           <1>       ; outb(VGAREG_ACTL_ADDRESS, ar_index);
12379                           <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
12380                           <1>       ;mov   dl, 0C0h
12381 00003DED 88E0            <1>       mov    al, ah ; ar_index
12382 00003DEF EE              <1>       out    dx, al
12383                           <1>
12384                           <1>       ; inb(VGAREG_ACTL_RESET);
12385                           <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
12386 00003DF0 B2DA            <1>       mov    dl, 0DAh
12387 00003DF2 EC              <1>       in     al, dx
12388                           <1>
12389                           <1>         ; for(i=0;i<=8;i++) {
12390 00003DF3 28C0            <1>       sub    al, al ; 0
12391                           <1>       ;;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
12392                           <1>       ;mov   dl, 0CEh
12393 00003DF5 B509            <1>       mov    ch, 9
12394                           <1> bfn_rvs_5:
12395                           <1>       ; outb(VGAREG_GRDC_ADDRESS,i)
12396                           <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
12397 00003DF7 B2CE            <1>       mov    dl, 0CEh
12398 00003DF9 EE              <1>       out    dx, al
12399                           <1>       ; outb(VGAREG_ACTL_READ_DATA)
12400 00003DFA 50              <1>       push   eax
12401                           <1>       ;mov   dx, VGAREG_GRDC_DATA ; 3CFh
12402                           <1>       ;mov   dl, 0CFh
12403 00003DFB FEC2            <1>       inc    dl
12404 00003DFD AC              <1>       lodsb  ; (9 bytes in loop)
12405 00003DFE EE              <1>       out    dx, al
12406 00003DFF 58              <1>       pop    eax
12407                           <1>       ;dec   dl
12408 00003E00 FEC0            <1>       inc    al  ; i++
12409 00003E02 FECD            <1>       dec    ch
12410 00003E04 75F1            <1>       jnz    short bfn_rvs_5
12411                           <1>
12412                           <1>       ; BX += 2; /* crtc_addr */     ; 3D4h
12413                           <1>         ; BX += 4; /* plane latches */ ; 0
12414 00003E06 83C606          <1>       add    esi, 6
12415 00003E09 56              <1>       push   esi ; *
12416                           <1>
12417                           <1>       ;outb(VGAREG_SEQU_ADDRESS, read_byte(ES, addr1)); addr1++;
12418                           <1>         ;outb(crtc_addr, read_byte(ES, addr1)); addr1++;
12419                           <1>         ;outb(VGAREG_GRDC_ADDRESS, read_byte(ES, addr1)); addr1++;
12420                           <1>         ;addr1++;
12421                           <1>         ;outb(crtc_addr - 0x4 + 0xa, read_byte(ES, addr1)); addr1++;
12422                           <1>
12423 00003E0A 89FE            <1>       mov    esi, edi ; start of state buffer
12424                           <1>
12425                           <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C7h
12426 00003E0C B2C7            <1>       mov    dl, 0C7h
12427 00003E0E AC              <1>       lodsb
12428 00003E0F EE              <1>       out    dx, al
12429                           <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12430 00003E10 B2D4            <1>       mov    dl, 0D4h
12431 00003E12 AC              <1>       lodsb
12432 00003E13 EE              <1>       out    dx, al
12433                           <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
12434 00003E14 B2CE            <1>         mov  dl, 0CEh
12435 00003E16 AC              <1>       lodsb
12436 00003E17 EE              <1>       out    dx, al
12437 00003E18 AC              <1>       lodsb  ; addr1++
12438                           <1>       ;mov   dx, VGAREG_VGA_WRITE_FEATURE_CTL ; 3DAh
12439 00003E19 B2DA            <1>       mov    dl, 0DAh
12440 00003E1B AC              <1>       lodsb
12441 00003E1C EE              <1>       out    dx, al
12442                           <1>
12443 00003E1D 5E              <1>       pop    esi ; *
```

```
12444                                   <1>
12445                                   <1>     ; (total 70 bytes are read above as controller hardware state)
12446                                   <1>
12447                                   <1> bfn_rvs_6:
12448                                   <1>     ; 13/01/2021
12449 00003E1E F6C102                   <1>     test   cl, 2
12450 00003E21 747D                     <1>     jz     short bfn_rvs_9
12451                                   <1>
12452                                   <1>     ; VIDEO BIOS DATA
12453                                   <1>     ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
12454                                   <1>     ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
12455                                   <1>
12456                                   <1>     ; if (CX & 2) {
12457                                   <1>     ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE, read_byte(ES, BX)); BX++;
12458                                   <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_NB_COLS, read_word(ES, BX)); BX += 2;
12459                                   <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, read_word(ES, BX)); BX += 2;
12460                                   <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS, read_word(ES, BX)); BX += 2;
12461                                   <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, read_byte(ES, BX)); BX++;
12462                                   <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, read_word(ES, BX)); BX += 2;
12463                                   <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL, read_byte(ES, BX)); BX++;
12464                                   <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES, read_byte(ES, BX)); BX++;
12465                                   <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL, read_byte(ES, BX)); BX++;
12466                                   <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE, read_word(ES, BX)); BX += 2;
12467                                   <1>       ;for(i=0;i<8;i++) {
12468                                   <1>       ;   write_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i, read_word(ES, BX));
12469                                   <1>       ;   BX += 2;
12470                                   <1>       ;}
12471                                   <1>       ;write_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START, read_word(ES, BX)); BX += 2;
12472                                   <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE, read_byte(ES, BX)); BX++;
12473                                   <1>       ;/* current font */
12474                                   <1>       ;write_word(0, 0x1f * 4, read_word(ES, BX)); BX += 2;
12475                                   <1>       ;write_word(0, 0x1f * 4 + 2, read_word(ES, BX)); BX += 2;
12476                                   <1>       ;write_word(0, 0x43 * 4, read_word(ES, BX)); BX += 2;
12477                                   <1>       ;write_word(0, 0x43 * 4 + 2, read_word(ES, BX)); BX += 2;
12478                                   <1>
12479                                   <1>     ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
12480                                   <1>     ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
12481                                   <1>
12482 00003E23 66AD                   <1>     lodsw  ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
12483                                   <1>     ; skip 3D4h check if it is already checked
12484 00003E25 F6C101                 <1>     test   cl, 1
12485 00003E28 7508                   <1>     jnz    short bfn_rvs_7
12486 00003E2A 663DD403               <1>     cmp    ax, 3D4h
12487 00003E2E 7402                   <1>     je     short bfn_rvs_7
12488 00003E30 F9                     <1>     stc
12489 00003E31 C3                     <1>     retn
12490                                   <1> bfn_rvs_7:
12491 00003E32 AC                     <1>     lodsb
12492 00003E33 A2[9A6E0000]           <1>     mov    [CRT_MODE], al ; Current video mode (0FFh for VESA VBE modes)
12493 00003E38 AC                     <1>     lodsb
12494 00003E39 A2[9B6E0000]           <1>     mov    [CRT_MODE_SET], al ; 29h for mode 03h ; TRDOS 386 feature only !
12495 00003E3E 66AD                   <1>     lodsw
12496 00003E40 66A3[1E120300]         <1>     mov    [video_mode], ax ; Current VESA VBE (SVGA, extended VGA) mode
12497 00003E46 66AD                   <1>     lodsw          ; (valid if [CRT_MODE] = 0FFh)
12498 00003E48 66A3[88950100]         <1>     mov    [CRT_LEN], ax ; page size (in bytes)
12499 00003E4E 66AD                   <1>     lodsw
12500 00003E50 66A3[0C890100]         <1>     mov    [CRT_START], ax ; video page start offset
12501 00003E56 AC                     <1>     lodsb
12502 00003E57 A2[9C6E0000]           <1>     mov    [CRT_COLS], al ; nbcols, characters per row
12503 00003E5C AC                     <1>     lodsb
12504 00003E5D A2[A26E0000]           <1>     mov    [VGA_ROWS], al ; nbrows, (character) rows per page (not rows-1)
12505 00003E62 AC                     <1>     lodsb
12506 00003E63 A2[9E6E0000]           <1>     mov    [CHAR_HEIGHT], al ; character font height (8 or 16 or 14)
12507 00003E68 AC                     <1>     lodsb
12508 00003E69 A2[9F6E0000]           <1>     mov    [VGA_VIDEO_CTL], al ; ROM BIOS DATA AREA Offset 87h
12509 00003E6E AC                     <1>     lodsb
12510 00003E6F A2[A06E0000]           <1>     mov    [VGA_SWITCHES], al ; feature bit switches
12511 00003E74 AC                     <1>     lodsb
12512 00003E75 A2[A16E0000]           <1>     mov    [VGA_MODESET_CTL], al ; basic mode set options
12513                                   <1>     ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
12514                                   <1>     ; (bochs/plex86 does not use and return those)
12515 00003E7A AC                     <1>     lodsb
12516 00003E7B A2[9D6E0000]           <1>     mov    [CRT_PALETTE], al ; current color palette ; TRDOS 386 feature only !
12517 00003E80 AC                     <1>     lodsb
12518 00003E81 A2[1E890100]           <1>     mov    [ACTIVE_PAGE], al ; current video page
12519 00003E86 66AD                   <1>     lodsw
12520 00003E88 66A3[B36E0000]         <1>     mov    [CURSOR_MODE], ax ; cursor type
12521                                   <1>     ;lodsd
12522                                   <1>     ;mov   [CURSOR_POSN], eax ; cursor position for video page 0 and 1
12523                                   <1>     ;lodsd
12524                                   <1>     ;mov   [CURSOR_POSN+4], eax ; cursor position for video page 2 and 3
12525                                   <1>     ;lodsd
12526                                   <1>     ;mov   [CURSOR_POSN+8], eax ; cursor position for video page 4 and 5
12527                                   <1>     ;lodsd
12528                                   <1>     ;mov   [CURSOR_POSN+12], eax ; cursor position for video page 6 and 7
12529 00003E8E B504                   <1>     mov    ch, 4
12530 00003E90 BF[0E890100]           <1>     mov    edi, CURSOR_POSN
12531                                   <1> bfn_rvs_8:
12532 00003E95 A5                     <1>     movsd
12533 00003E96 FECD                   <1>     dec    ch
12534 00003E98 75FB                   <1>     jnz    short bfn_rvs_8
12535                                   <1>     ; (font addr) protected mode address in kernel's/system memory space
12536                                   <1>     ; (not accessable/meaningful address value by user)
12537 00003E9A AD                     <1>     lodsd
12538 00003E9B A3[9A950100]           <1>     mov    [VGA_INT43H], eax ; VGA current (default) font address
12539                                   <1>
12540                                   <1>     ; (total 40 bytes are read&written above as BIOS data state)
12541                                   <1> bfn_rvs_9:
12542                                   <1>     ; 13/01/2021
12543 00003EA0 F6C104                 <1>     test   cl, 4
12544 00003EA3 7422                   <1>     jz     short bfn_rvs_11
12545                                   <1>
12546                                   <1>     ;BX++;
12547                                   <1>       ;v = read_byte(ES, BX); BX++;
12548                                   <1>       ;outb(VGAREG_PEL_MASK, read_byte(ES, BX)); BX++;
```

```
12549                                  <1>         ;// Set the whole dac always, from 0
12550                                  <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
12551                                  <1>         ;for(i=0;i<256*3;i++) {
12552                                  <1>         ;   outb(VGAREG_DAC_DATA, read_byte(ES, BX)); BX++;
12553                                  <1>         ;}
12554                                  <1>         ;BX++;
12555                                  <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS, v);
12556                                  <1>
12557                                  <1>       ; /* read/write mode dac */
12558 00003EA5 AC                     <1>       lodsb  ; skip ; VGAREG_DAC_STATE
12559 00003EA6 AC                     <1>       lodsb
12560 00003EA7 88C4                   <1>       mov    ah, al ; * ; v
12561 00003EA9 AC                     <1>       lodsb
12562 00003EAA 66BAC603               <1>       mov    dx, VGAREG_PEL_MASK  ; 3C6h
12563 00003EAE EE                     <1>       out    dx, al
12564                                  <1>       ;// Set the whole dac always, from 0
12565 00003EAF 30C0                   <1>       xor    al, al ; 0
12566                                  <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
12567 00003EB1 B2C8                   <1>       mov    dl, 0C8h
12568 00003EB3 EE                     <1>       out    dx, al
12569                                  <1>
12570 00003EB4 51                     <1>       push   ecx ; 22/01/2021
12571                                  <1>       ;for(i=0;i<256*3;i++) {
12572 00003EB5 B900030000             <1>       mov    ecx, 256*3 ; 768 bytes
12573                                  <1>       ;mov   dx, VGAREG_DAC_DATA ; 3C9h
12574                                  <1>       ;mov   dl, 0C9h
12575 00003EBA FEC2                   <1>       inc    dl ; dx = 3C9h
12576                                  <1> bfn_rvs_10:
12577 00003EBC AC                     <1>       lodsb
12578 00003EBD EE                     <1>       out    dx, al
12579 00003EBE E2FC                   <1>       loop   bfn_rvs_10
12580 00003EC0 59                     <1>       pop    ecx ; 22/01/2021
12581                                  <1>
12582                                  <1>       ; /* color select register */
12583 00003EC1 AC                     <1>       lodsb   ; skip
12584                                  <1>
12585 00003EC2 88E0                   <1>       mov    al, ah ; * ; v
12586                                  <1>
12587                                  <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
12588                                  <1>       ;mov   dl, 0C8h
12589 00003EC4 FECA                   <1>       dec    dl ; dx  = 3C8h
12590 00003EC6 EE                     <1>       out    dx, al ; * ; v
12591                                  <1>
12592                                  <1>       ; (total 772 bytes are read above as DAC state)
12593                                  <1> bfn_rvs_11:
12594 00003EC7 C3                     <1>       retn
12595                                  <1>
12596                                  <1> vbe_biosfn_restore_video_state:
12597                                  <1>       ; 23/01/2021
12598                                  <1>       ; 13/01/2021 (TRDOS 386 v2.0.3)
12599                                  <1>       ; (vbe.c)
12600                                  <1>
12601                                  <1>       ; modified registers: eax, edx, esi, ch
12602                                  <1>
12603                                  <1>       ; input: esi = state buffer address
12604                                  <1>       ; output:
12605                                  <1>       ;       VBE DISPI register contents will be restored
12606                                  <1>       ;       (18 bytes, 9 words)
12607                                  <1>
12608                                  <1>       ; enable = read_word(ES, BX);
12609                                  <1>       ; BX += 2;
12610                                  <1>       ;
12611                                  <1>       ; if (!(enable & VBE_DISPI_ENABLED)) {
12612                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
12613                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
12614                                  <1>       ; } else {
12615                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_XRES);
12616                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12617                                  <1>         ;   BX += 2;
12618                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_YRES);
12619                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12620                                  <1>         ;   BX += 2;
12621                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_BPP);
12622                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12623                                  <1>         ;   BX += 2;
12624                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
12625                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
12626                                  <1>       ;
12627                                  <1>         ;  for(i = VBE_DISPI_INDEX_BANK; i <= VBE_DISPI_INDEX_Y_OFFSET; i++)
12628                                  <1>       ;    {
12629                                  <1>         ;     outw(VBE_DISPI_IOPORT_INDEX, i);
12630                                  <1>         ;     outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12631                                  <1>         ;     BX += 2;
12632                                  <1>         ;    }
12633                                  <1>         ; }
12634                                  <1>
12635 00003EC8 66AD                   <1>       lodsw  ; enable (status, enabled=1, disabled=0)
12636 00003ECA 66BACE01               <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12637                                  <1>       ; 23/01/2021
12638 00003ECE 6683E001               <1>       and    ax, 1 ; VBE_DISPI_ENABLED
12639 00003ED2 750B                   <1>       jnz    short vbe_bfn_rvs_1
12640                                  <1>       ; ax = 0
12641                                  <1>       ; VBE_DISPI_DISABLED
12642                                  <1> vbe_bfn_rvs_0:
12643                                  <1>       ; enable (disable) dispi
12644                                  <1>       ; dx = 01CEh ; VBE_DISPI_IOPORT_INDEX
12645                                  <1>       ; ah = 0
12646 00003ED4 50                     <1>       push   eax
12647 00003ED5 B004                   <1>       mov    al, 04h      ; VBE_DISPI_INDEX_ENABLE
12648 00003ED7 66EF                   <1>       out    dx, ax
12649                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12650 00003ED9 FEC2                   <1>       inc    dl
12651 00003EDB 58                     <1>       pop    eax
12652 00003EDC 66EF                   <1>       out    dx, ax ; enable (or disable)
12653 00003EDE C3                     <1>       retn
```

```
                        <1> vbe_bfn_rvs_1:
                        <1>       ; VBE_DISPI_ENABLED
                        <1>
                        <1>       ; from VBE_DISPI_INDEX_XRES
                        <1>       ;   to VBE_DISPI_INDEX_BPP
                        <1>
00003EDF B503           <1>       mov    ch, 3
00003EE1 28C0           <1>       sub    al, al ; 0 ; VBE_DISPI_INDEX_XRES - 1
                        <1>       ; ax = 0
                        <1>
00003EE3 E80B000000     <1>       call   vbe_bfn_rvs_2
                        <1>
                        <1>         ;outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
                        <1>       ;outw(VBE_DISPI_IOPORT_DATA, enable);
                        <1>
                        <1>       ; 23/01/2021
00003EE8 B001           <1>       mov    al, 1 ; VBE_DISPI_ENABLED
                        <1>       ; ax = 1
00003EEA E8E5FFFFFF     <1>       call   vbe_bfn_rvs_0
                        <1>
                        <1>       ; from VBE_DISPI_INDEX_BANK
                        <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
                        <1>
00003EEF B505           <1>       mov    ch, 5
                        <1>       ; 23/01/2021
00003EF1 B004           <1>       mov    al, 4  ; VBE_DISPI_INDEX_BANK - 1
                        <1>       ; ax = 4
                        <1> vbe_bfn_rvs_2:
00003EF3 FEC0           <1>       inc    al ; from VBE_DISPI_INDEX_XRES
                        <1>                 ; to VBE_DISPI_INDEX_BPP
                        <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                        <1>       ;mov   dl, 0CEh
00003EF5 66EF           <1>       out    dx, ax
00003EF7 50             <1>       push   eax
                        <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
00003EF8 FEC2           <1>       inc    dl ; 1CFh
00003EFA 66AD           <1>       lodsw
00003EFC 66EF           <1>       out    dx, ax
00003EFE 58             <1>       pop    eax
00003EFF FECA           <1>       dec    dl ; 1CEh
00003F01 FECD           <1>       dec    ch
00003F03 75EE           <1>       jnz    short vbe_bfn_rvs_2
00003F05 C3             <1>       retn
                        <1>
                        <1> ; ---------------------------------------------------------
                        <1>
                        <1> dispi_set_enable:
                        <1>       ; 23/11/2020
                        <1>       ; Input:
                        <1>       ;      ax = VBE_DISPI_ENABLED = 1
                        <1>       ;           or VBE_DISPI_DISABLED = 0
                        <1>       ;
                        <1>       ; Modified registers: none
                        <1>
                        <1>       ;push  edx
                        <1>       ;push  eax
                        <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                        <1>       ;mov   ax, 04h        ; VBE_DISPI_INDEX_ENABLE
                        <1>       ;out   dx, ax
                        <1>       ;pop   eax
                        <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                        <1>       ;;mov  dl, 0CFh
                        <1>       ;inc   dl
                        <1>       ;out   dx, ax
                        <1>       ;pop   edx
                        <1>       ;retn
                        <1>
                        <1>       ; 25/11/2020
                        <1>       ; Modified registers: edx
                        <1>       ;push  edx
00003F06 66BA0400       <1>       mov    dx, 04h ; VBE_DISPI_INDEX_ENABLE
                        <1>       ;call  dispi_set_parms
                        <1>       ;pop   edx
                        <1>       ;retn
                        <1>       ;;jmp  short dispi_set_parms
                        <1>
                        <1> dispi_set_parms:
                        <1>       ; 25/11/2020
                        <1>       ; Input:
                        <1>       ;      ax = data
                        <1>       ;      dx = vbe dispi register index
                        <1>       ;
                        <1>       ; Modified registers: edx
                        <1>
00003F0A 50             <1>       push   eax
00003F0B 6689D0         <1>       mov    ax, dx
00003F0E 66BACE01       <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003F12 66EF           <1>       out    dx, ax
00003F14 58             <1>       pop    eax
                        <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                        <1>       ;mov   dl, 0CFh
00003F15 FEC2           <1>       inc    dl
00003F17 66EF           <1>       out    dx, ax
00003F19 C3             <1>       retn
                        <1>
                        <1> dispi_set_bpp:
                        <1>       ; 25/11/2020
                        <1>       ; Input:
                        <1>       ;      ax = Bits per pixel value
                        <1>       ;           (8,16,24,32)
                        <1>       ;
                        <1>       ; Modified registers: none
                        <1>
                        <1>       ;push  edx
                        <1>       ;push  eax
```

```
12759                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12760                                  <1>       ;mov   ax, 03h        ; VBE_DISPI_INDEX_BPP
12761                                  <1>       ;out   dx, ax
12762                                  <1>       ;pop   eax
12763                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12764                                  <1>       ;;mov  dl, 0CFh
12765                                  <1>       ;inc   dl
12766                                  <1>       ;out   dx, ax
12767                                  <1>       ;pop   edx
12768                                  <1>       ;retn
12769                                  <1>
12770                                  <1>       ; 25/11/2020
12771                                  <1>       ; Modified registers: edx
12772                                  <1>       ;push  edx
12773 00003F1A 66BA0300               <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
12774                                  <1>       ;call  dispi_set_parms
12775                                  <1>       ;pop   edx
12776                                  <1>       ;retn
12777 00003F1E EBEA                   <1>       jmp    short dispi_set_parms
12778                                  <1>
12779                                  <1> dispi_set_xres:
12780                                  <1>       ; 25/11/2020
12781                                  <1>       ; Input:
12782                                  <1>       ;      ax = X resolution (screen witdh)
12783                                  <1>       ;           (320,640,800,1024,1280,1920)
12784                                  <1>       ;
12785                                  <1>       ; Modified registers: none
12786                                  <1>
12787                                  <1>       ;push  edx
12788                                  <1>       ;push  eax
12789                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12790                                  <1>       ;mov   ax, 01h        ; VBE_DISPI_INDEX_XRES
12791                                  <1>       ;out   dx, ax
12792                                  <1>       ;pop   eax
12793                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12794                                  <1>       ;;mov  dl, 0CFh
12795                                  <1>       ;inc   dl
12796                                  <1>       ;out   dx, ax
12797                                  <1>       ;pop   edx
12798                                  <1>       ;retn
12799                                  <1>
12800                                  <1>       ; 25/11/2020
12801                                  <1>       ; Modified registers: edx
12802                                  <1>       ;push  edx
12803 00003F20 66BA0100               <1>       mov    dx, 01h ; VBE_DISPI_INDEX_XRES
12804                                  <1>       ;call  dispi_set_parms
12805                                  <1>       ;pop   edx
12806                                  <1>       ;retn
12807 00003F24 EBE4                   <1>       jmp    short dispi_set_parms
12808                                  <1>
12809                                  <1> dispi_set_yres:
12810                                  <1>       ; 25/11/2020
12811                                  <1>       ; Input:
12812                                  <1>       ;      ax = Y resolution (screen height)
12813                                  <1>       ;           (200,400,600,720,768,1080)
12814                                  <1>       ;
12815                                  <1>       ; Modified registers: none
12816                                  <1>
12817                                  <1>       ;push  edx
12818                                  <1>       ;push  eax
12819                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12820                                  <1>       ;mov   ax, 02h        ; VBE_DISPI_INDEX_YRES
12821                                  <1>       ;out   dx, ax
12822                                  <1>       ;pop   eax
12823                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12824                                  <1>       ;;mov  dl, 0CFh
12825                                  <1>       ;inc   dl
12826                                  <1>       ;out   dx, ax
12827                                  <1>       ;pop   edx
12828                                  <1>       ;retn
12829                                  <1>
12830                                  <1>       ; 25/11/2020
12831                                  <1>       ; Modified registers: edx
12832                                  <1>       ;push  edx
12833 00003F26 66BA0200               <1>       mov    dx, 02h  ; VBE_DISPI_INDEX_YRES
12834                                  <1>       ;call  dispi_set_parms
12835                                  <1>       ;pop   edx
12836                                  <1>       ;retn
12837 00003F2A EBDE                   <1>       jmp    short dispi_set_parms
12838                                  <1>
12839                                  <1> dispi_set_bank:
12840                                  <1>       ; 25/11/2020
12841                                  <1>       ; Input:
12842                                  <1>       ;      ax = video memory bank number
12843                                  <1>       ;
12844                                  <1>       ; Modified registers: none
12845                                  <1>
12846                                  <1>       ;push  edx
12847                                  <1>       ;push  eax
12848                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12849                                  <1>       ;mov   ax, 05h        ; VBE_DISPI_INDEX_BANK
12850                                  <1>       ;out   dx, ax
12851                                  <1>       ;pop   eax
12852                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12853                                  <1>       ;;mov  dl, 0CFh
12854                                  <1>       ;inc   dl
12855                                  <1>       ;out   dx, ax
12856                                  <1>       ;pop   edx
12857                                  <1>       ;retn
12858                                  <1>
12859                                  <1>       ; 25/11/2020
12860                                  <1>       ; Modified registers: edx
12861                                  <1>       ;push  edx
12862 00003F2C 66BA0500               <1>       mov    dx, 05h  ; VBE_DISPI_INDEX_BANK
12863                                  <1>       ;call  dispi_set_parms
```

```
12864                                  <1>       ;pop   edx
12865                                  <1>       ;retn
12866 00003F30 EBD8                    <1>       jmp    short dispi_set_parms
12867                                  <1>
12868                                  <1> dispi_get_enable:
12869                                  <1>       ; 27/11/2020
12870                                  <1>       ; Input:
12871                                  <1>       ;      none
12872                                  <1>       ; Output:
12873                                  <1>       ;      ax = vbe dispi status
12874                                  <1>       ;
12875                                  <1>       ; Modified registers: eax
12876                                  <1>
12877                                  <1>       ;push  edx
12878                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12879                                  <1>       ;mov   ax, 04h        ; VBE_DISPI_INDEX_ENABLE
12880                                  <1>       ;out   dx, ax
12881                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12882                                  <1>       ;;mov  dl, 0CFh
12883                                  <1>       ;inc   dl
12884                                  <1>       ;in    ax, dx
12885                                  <1>       ;pop   edx
12886                                  <1>       ;retn
12887                                  <1>
12888                                  <1>       ; 27/11/2020
12889                                  <1>       ; Modified registers: eax, edx
12890                                  <1>       ;push  edx
12891 00003F32 66B80400                <1>       mov    ax, 04h ; VBE_DISPI_INDEX_ENABLE
12892                                  <1>       ;call  dispi_get_parms
12893                                  <1>       ;pop   edx
12894                                  <1>       ;retn
12895                                  <1>       ;;jmp  short dispi_get_parms
12896                                  <1>
12897                                  <1> dispi_get_parms:
12898                                  <1>       ; 25/11/2020
12899                                  <1>       ; Input:
12900                                  <1>       ;      ax = vbe dispi register index
12901                                  <1>       ; output:
12902                                  <1>       ;      ax = data
12903                                  <1>       ;
12904                                  <1>       ; Modified registers: eax, edx
12905                                  <1>
12906 00003F36 66BACE01                <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12907 00003F3A 66EF                    <1>       out    dx, ax
12908                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12909                                  <1>       ;mov   dl, 0CFh
12910 00003F3C FEC2                    <1>       inc    dl
12911 00003F3E 66ED                    <1>       in     ax, dx
12912 00003F40 C3                      <1>       retn
12913                                  <1>
12914                                  <1> vga_compat_setup:
12915                                  <1>       ; 26/11/2020
12916                                  <1>       ; 25/11/2020
12917                                  <1>       ; VGA compatibility setup
12918                                  <1>       ; (vbe.c, 02/01/2020, vruppert)
12919                                  <1>       ;
12920                                  <1>       ; Input:
12921                                  <1>       ;      none
12922                                  <1>       ;
12923                                  <1>       ; Modified registers: eax, edx
12924                                  <1>
12925                                  <1>       ; 26/11/2020
12926                                  <1>       ;push  eax
12927                                  <1>       ;push  edx
12928                                  <1>
12929                                  <1>       ; set CRT X resolution
12930 00003F41 66BACE01                <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
12931 00003F45 66B80100                <1>       mov    ax, 01h  ; VBE_DISPI_INDEX_XRES
12932 00003F49 66EF                    <1>       out    dx, ax
12933                                  <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
12934 00003F4B FEC2                    <1>       inc    dl
12935 00003F4D 66ED                    <1>       in     ax, dx
12936 00003F4F 50                      <1>       push   eax
12937 00003F50 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
12938 00003F54 66B81100                <1>       mov    ax, 0011h ; Vertical retrace end register
12939 00003F58 66EF                    <1>       out    dx, ax
12940                                  <1>       ;pop   eax
12941                                  <1>       ;push  eax
12942 00003F5A 8B0424                  <1>       mov    eax, [esp]
12943 00003F5D 66C1E803                <1>       shr    ax, 3 ; / 8 for pixel to character
12944 00003F61 6648                    <1>       dec    ax  ; - 1 (EGA or VGA?)
12945 00003F63 88C4                    <1>       mov    ah, al
12946 00003F65 B001                    <1>       mov    al, 01h ; Horizontal display end register
12947 00003F67 66EF                    <1>       out    dx, ax
12948 00003F69 58                      <1>       pop    eax
12949                                  <1>
12950 00003F6A E8B0000000              <1>       call   vga_set_virt_width
12951                                  <1>
12952                                  <1>       ; set CRT Y resolution
12953 00003F6F 66BACE01                <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
12954 00003F73 66B80200                <1>       mov    ax, 02h  ; VBE_DISPI_INDEX_YRES
12955 00003F77 66EF                    <1>       out    dx, ax
12956                                  <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
12957 00003F79 FEC2                    <1>       inc    dl
12958 00003F7B 66ED                    <1>       in     ax, dx
12959 00003F7D 50                      <1>       push   eax
12960 00003F7E 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
12961 00003F82 88C4                    <1>       mov    ah, al
12962 00003F84 B012                    <1>       mov    al, 12h ; Vertical display end register
12963 00003F86 66EF                    <1>       out    dx, ax
12964 00003F88 58                      <1>       pop    eax
12965 00003F89 B007                    <1>       mov    al, 07h      ; Overflow register
12966 00003F8B EE                      <1>       out    dx, al
12967 00003F8C 6642                    <1>       inc    dx
12968 00003F8E EC                      <1>       in     al, dx ; read overflow register
```

```
12969 00003F8F 24BD              <1>       and   al, 0BDh ; clear VDE 9th and 10th bits
12970 00003F91 F6C401            <1>       test  ah, 01h
12971 00003F94 7402              <1>       jz    short bit8_clear
12972 00003F96 0C02              <1>       or    al, 02h ; VDE 9th bit (bit 8) in bit 1
12973                            <1> bit8_clear:
12974 00003F98 F6C402            <1>       test  ah, 02h
12975 00003F9B 7402              <1>       jz    short bit9_clear
12976 00003F9D 0C40              <1>       or    al, 40h ; VDE 10th bit (bit 9) in bit 6
12977                            <1> bit9_clear:
12978 00003F9F EE                <1>       out   dx, al
12979                            <1>
12980                            <1>       ; other settings
12981 00003FA0 66BAD403          <1>       mov   dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
12982 00003FA4 66B80900          <1>       mov   ax, 0009h ; Maximum scan line register
12983 00003FA8 66EF              <1>       out   dx, ax ; Reset
12984 00003FAA B017              <1>       mov   al, 17h ; Mode control register
12985 00003FAC EE                <1>       out   dx, al
12986                            <1>       ;mov  dx, 3D5h ; VGAREG_VGA_CRTC_DATA
12987 00003FAD FEC2              <1>       inc   dl
12988 00003FAF EC                <1>       in    al, dx ; Read mode control register
12989 00003FB0 0C03              <1>       or    al, 03h ; Set SRS and CMS bits
12990 00003FB2 EE                <1>       out   dx, al
12991 00003FB3 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
12992 00003FB7 EC                <1>       in    al, dx  ; clear flip-flop
12993 00003FB8 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
12994 00003FBC B010              <1>       mov   al, 10h      ; Mode control register
12995 00003FBE EE                <1>       out   dx, al
12996                            <1>       ;mov  dx, 3C1h ; VGAREG_ACTL_READ_DATA
12997 00003FBF FEC2              <1>       inc   dl
12998 00003FC1 EC                <1>       in    al, dx
12999 00003FC2 0C01              <1>       or    al, 01h ; select graphics mode
13000                            <1>       ;mov  dx, 3C0h ; VGAREG_ACTL_ADDRESS
13001 00003FC4 FECA              <1>       dec   dl
13002 00003FC6 EE                <1>       out   dx, al ; Write to mode control register
13003 00003FC7 B020              <1>       mov   al, 20h ; Palette RAM <-> display memory
13004 00003FC9 EE                <1>       out   dx, al ; Write to attribute addr register
13005 00003FCA 66BACE03          <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
13006 00003FCE 66B80605          <1>       mov   ax, 0506h ; Misc. register, graph, mm 1
13007 00003FD2 66EF              <1>       out   dx, ax
13008 00003FD4 66BAC403          <1>       mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
13009 00003FD8 66B8020F          <1>       mov   ax, 0F02h ; Map mask register, all planes
13010 00003FDC 66EF              <1>       out   dx, ax
13011                            <1>
13012                            <1>       ; settings for >= 8bpp
13013                            <1>
13014                            <1>       ;mov  dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
13015                            <1>       ;mov  ax, 03h ; VBE_DISPI_INDEX_BPP
13016                            <1>       ;out  dx, ax
13017                            <1>       ;;mov dx, 1CFh ; VBE_DISPI_IOPORT_DATA
13018                            <1>       ;inc  dl
13019                            <1>       ;in   ax, dx
13020                            <1>       ;cmp  al, 08h  ; < 8 bits per pixel
13021                            <1>       ;jb   short vga_compat_end
13022                            <1>
13023 00003FDE 66BAD403          <1>       mov   dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
13024 00003FE2 B014              <1>       mov   al, 14h  ; Underline location register
13025 00003FE4 EE                <1>       out   dx, al
13026                            <1>       ;mov  dx, 3D5h ; VGAREG_VGA_CRTC_DATA
13027 00003FE5 FEC2              <1>       inc   dl
13028 00003FE7 EC                <1>       in    al, dx
13029 00003FE8 0C40              <1>       or    al, 40h       ; enable double word mode
13030 00003FEA EE                <1>       out   dx, al
13031 00003FEB 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
13032 00003FEF EC                <1>       in    al, dx  ; clear flip-flop
13033 00003FF0 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
13034 00003FF4 B010              <1>       mov   al, 10h       ; Mode control register
13035 00003FF6 EE                <1>       out   dx, al
13036                            <1>       ;mov  dx, 3C1h ; VGAREG_ACTL_READ_DATA
13037 00003FF7 FEC2              <1>       inc   dl
13038 00003FF9 EC                <1>       in    al, dx
13039 00003FFA 0C40              <1>       or    al, 40h  ; Pixel clock select is 1
13040                            <1>       ;mov  dx, 3C0h ; VGAREG_ACTL_ADDRESS
13041 00003FFC FECA              <1>       dec   dl
13042 00003FFE EE                <1>       out   dx, al  ; update mode control reggister
13043 00003FFF B020              <1>       mov   al, 20h       ; select display memory as PAS
13044 00004001 EE                <1>       out   dx, al
13045 00004002 66BAC403          <1>       mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
13046 00004006 B004              <1>       mov   al, 04h      ; Memory mode register
13047 00004008 EE                <1>       out   dx, al
13048                            <1>       ;mov  dx, 3C5h ; VGAREG_SEQU_DATA
13049 00004009 FEC2              <1>       inc   dl
13050 0000400B EC                <1>       in    al, dx
13051 0000400C 0C08              <1>       or    al, 08h      ; enable chain four
13052 0000400E EE                <1>       out   dx, al
13053 0000400F 66BACE03          <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
13054 00004013 B005              <1>       mov   al, 05h       ; Mode register
13055 00004015 EE                <1>       out   dx, al
13056                            <1>       ;mov  dx, 3CFh ; VGAREG_GRDC_DATA
13057 00004016 FEC2              <1>       inc   dl
13058 00004018 EC                <1>       in    al, dx
13059 00004019 249F              <1>       and   al, 9Fh       ; clear shift register
13060 0000401B 0C40              <1>       or    al, 40h  ; set shift register to 2
13061 0000401D EE                <1>       out   dx, al
13062                            <1>
13063                            <1> vga_compat_end:
13064                            <1>       ;pop  edx
13065                            <1>       ;pop  eax
13066 0000401E C3                <1>       retn
13067                            <1>
13068                            <1> vga_set_virt_width:
13069                            <1>       ; 27/11/2020
13070                            <1>       ; 25/11/2020
13071                            <1>       ; (vbe.c, 02/01/2020, vruppert)
13072                            <1>       ;
13073                            <1>       ; Input:
```

```
13074                                   <1>       ;      AX = resolution (screen width)
13075                                   <1>       ;
13076                                   <1>       ; Modified registers: eax, edx
13077                                   <1>
13078                                   <1>       ;;push ebx
13079                                   <1>       ;push  edx
13080                                   <1>       ;push  eax
13081                                   <1>       ;mov   ebx, eax
13082                                   <1>       ;call  dispi_get_bpp ; bits per pixel
13083                                   <1>       ;cmp   al, 4
13084                                   <1>       ;ja    short set_width_svga  ; 8, 16, 24, 32
13085                                   <1>       ;shr   bx, 1
13086                                   <1> ;set_width_svga:
13087                                   <1>       ;shr   bx, 3
13088                                   <1>       ;mov   eax, [esp]
13089 0000401F 66C1E803                <1>       shr    ax, 3  ; / 8, bytes per row
13090 00004023 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
13091                                   <1>       ;mov   ah, bl ;
13092 00004027 88C4                    <1>       mov    ah, al ; width in bytes
13093 00004029 B013                    <1>       mov    al, 13h      ; offset register
13094 0000402B 66EF                    <1>       out    dx, ax ; index (3D4h) and data (3D5h)
13095                                   <1>       ;pop   eax
13096                                   <1>       ;pop   edx
13097                                   <1>       ;;pop  ebx
13098 0000402D C3                      <1>       retn
13099                                   <1>
13100                                   <1> ; 24/11/2020
13101                                   <1>
13102                                   <1> struc bmi ; BOCHS/PLEX86 MODE INFO structure/table
13103 00000000 ????                    <1>  .mode:        resw 1
13104 00000002 ????                    <1>  .width:  resw 1
13105 00000004 ????                    <1>  .height: resw 1
13106 00000006 ????                    <1>  .depth:  resw 1
13107                                   <1>  .size:
13108                                   <1> endstruc
13109                                   <1>
13110                                   <1> ; 24/11/2020
13111                                   <1> struc MODEINFO
13112 00000000 ????                    <1>  .mode:                   resw 1  ; 1XXh
13113 00000002 ????                    <1>  .ModeAttributes:  resw 1
13114 00000004 ??                      <1>  .WinAAttributes:  resb 1
13115 00000005 ??                      <1>  .WinBAttributes:  resb 1 ; = 0
13116 00000006 ????                    <1>  .WinGranularity:  resw 1
13117 00000008 ????                    <1>  .WinSize:         resw 1
13118 0000000A ????                    <1>  .WinASegment:            resw 1
13119 0000000C ????                    <1>  .WinBSegment:            resw 1 ; = 0
13120 0000000E ????????                <1>  .WinFuncPtr:             resd 1 ; = 0
13121 00000012 ????                    <1>  .BytesPerScanLine:       resw 1
13122 00000014 ????                    <1>  .XResolution:            resw 1
13123 00000016 ????                    <1>  .YResolution:            resw 1
13124 00000018 ??                      <1>  .XCharSize:       resb 1
13125 00000019 ??                      <1>  .YCharSize:       resb 1
13126 0000001A ??                      <1>  .NumberOfPlanes:  resb 1
13127 0000001B ??                      <1>  .BitsPerPixel:           resb 1
13128 0000001C ??                      <1>  .NumberOfBanks:   resb 1
13129 0000001D ??                      <1>  .MemoryModel:            resb 1
13130 0000001E ??                      <1>  .BankSize:        resb 1 ; = 0
13131 0000001F ??                      <1>  .NumberOfImagePages:     resb 1
13132 00000020 ??                      <1>  .Reserved_page:   resb 1 ; = 0
13133 00000021 ??                      <1>  .RedMaskSize:            resb 1
13134 00000022 ??                      <1>  .RedFieldPosition:       resb 1
13135 00000023 ??                      <1>  .GreenMaskSize:   resb 1
13136 00000024 ??                      <1>  .GreenFieldPosition:     resb 1
13137 00000025 ??                      <1>  .BlueMaskSize:           resb 1
13138 00000026 ??                      <1>  .BlueFieldPosition:      resb 1
13139 00000027 ??                      <1>  .RsvdMaskSize:           resb 1
13140 00000028 ??                      <1>  .RsvdFieldPosition:      resb 1
13141 00000029 ??                      <1>  .DirectColorModeInfo:    resb 1
13142 0000002A ????????                <1>  .PhysBasePtr:            resd 1
13143 0000002E ????????                <1>  .OffScreenMemOffset:     resd 1 ; = 0
13144 00000032 ????                    <1>  .OffScreenMemSize:       resw 1 ; = 0
13145 00000034 ????                    <1>  .LinBytesPerScanLine:    resw 1
13146 00000036 ??                      <1>  .BnkNumberOfPages:       resb 1
13147 00000037 ??                      <1>  .LinNumberOfPages:       resb 1
13148 00000038 ??                      <1>  .LinRedMaskSize:  resb 1
13149 00000039 ??                      <1>  .LinRedFieldPosition1: resb 1
13150 0000003A ??                      <1>  .LinGreenMaskSize1:      resb 1
13151 0000003B ??                      <1>  .LinGreenFieldPosition:resb 1
13152 0000003C ??                      <1>  .LinBlueMaskSize:        resb 1
13153 0000003D ??                      <1>  .LinBlueFieldPosition:   resb 1
13154 0000003E ??                      <1>  .LinRsvdMaskSize:        resb 1
13155 0000003F ??                      <1>  .LinRsvdFieldPosition:   resb 1
13156 00000040 ????????                <1>  .MaxPixelClock:   resd 1 ; = 0
13157                                   <1> .size:
13158                                   <1> endstruc
13159                                   <1>
13160                                   <1> ; 10/12/2020
13161                                   <1> struc LFBINFO
13162 00000000 ????                    <1>  .mode:                   resw 1  ; 1XXh
13163 00000002 ????????                <1>  .LFB_addr:        resd 1
13164 00000006 ????????                <1>  .LFB_size:        resd 1
13165 0000000A ????                    <1>  .X_res:           resw 1
13166 0000000C ????                    <1>  .Y_res:           resw 1
13167 0000000E ??                      <1>  .bpp:                    resb 1
13168 0000000F ??                      <1>  .reserved:        resb 1
13169                                   <1> .size:      ; 16 bytes
13170                                   <1> endstruc
13171                                   <1>
13172                                   <1> set_mode_info_list:
13173                                   <1>       ; 14/12/2020
13174                                   <1>       ; 11/12/2020
13175                                   <1>       ; 24/11/2020
13176                                   <1>       ; (vbetables-gen.c)
13177                                   <1>       ; Input:
13178                                   <1>       ;      BX = VBE mode (including bochs special modes)
```

```
13179                                  <1>       ; Output:
13180                                  <1>       ;      ;;EAX = MODE_INFO_LIST address
13181                                  <1>       ;      EAX = 0 ; 11/12/2020
13182                                  <1>       ;      ESI = MODE_INFO_LIST address ; 11/12/2020
13183                                  <1>       ;      (if mode is not found, ESI = 0)
13184                                  <1>       ;
13185                                  <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
13186                                  <1>
13187 0000402E BE[26720000]            <1>       mov    esi, b_vbe_modes ; bochs mode info base table
13188 00004033 BF[3A120300]            <1>       mov    edi, MODE_INFO_LIST ; mode info list (4F01h)
13189                                  <1> sml_0:
13190 00004038 66AD                    <1>       lodsw
13191 0000403A 6639D8                  <1>       cmp    ax, bx ; is mode number same ?
13192 0000403D 7410                    <1>       je     short sml_1 ; yes
13193 0000403F AD                      <1>       lodsd
13194 00004040 66AD                    <1>       lodsw
13195 00004042 81FE[E6720000]          <1>       cmp    esi, end_of_b_vbe_modes
13196 00004048 72EE                    <1>       jb     short sml_0
13197                                  <1>       ; not found
13198 0000404A 31C0                    <1>       xor    eax, eax ; 0
13199                                  <1>       ; 11/12/2020
13200 0000404C 31F6                    <1>       xor    esi, esi
13201 0000404E C3                      <1>       retn
13202                                  <1> sml_1:
13203 0000404F 66AB                    <1>       stosw  ; mode
13204 00004051 AD                      <1>       lodsd  ; width, height
13205                                  <1>       ; 14/12/2020
13206 00004052 89C1                    <1>       mov    ecx, eax
13207 00004054 50                      <1>       push   eax ; ***
13208 00004055 29C0                    <1>       sub    eax, eax ; clear high word of eax
13209 00004057 66AD                    <1>       lodsw  ; depth
13210 00004059 50                      <1>       push   eax ; **
13211                                  <1>
13212                                  <1>       ;add  al, 7 ; only for 15 bit colors (not used here)
13213 0000405A C0E803                  <1>       shr    al, 3 ; / 8
13214                                  <1>       ; 14/12/2020
13215 0000405D 66F7E1                  <1>       mul    cx  ; pitch = width * ((depth+7)/8)
13216                                  <1>              ; ax = pitch
13217 00004060 50                      <1>       push   eax ; * ; high word of eax = 0
13218 00004061 C1E910                  <1>       shr    ecx, 16
13219                                  <1>       ;mul   cx
13220                                  <1>       ;mov   cx, ax
13221 00004064 31D2                    <1>       xor    edx, edx  ; clear high word of edx
13222 00004066 F7E1                    <1>       mul    ecx ; height * pitch
13223 00004068 89C1                    <1>       mov    ecx, eax
13224 0000406A B800000001              <1>       mov    eax, VBE_DISPI_TOTAL_VIDEO_MEMORY_MB * 1024 * 1024
13225 0000406F F7F1                    <1>       div    ecx
13226                                  <1>              ; eax = pages = vram_size / (height*pitch)
13227                                  <1>
13228                                  <1>       ;mov   cx, ax
13229 00004071 89C1                    <1>       mov    ecx, eax ; pages
13230                                  <1>
13231 00004073 66B89B00                <1>       mov    ax, MODE_ATTRIBUTES
13232 00004077 66AB                    <1>       stosw  ; ModeAttributes
13233 00004079 B007                    <1>       mov    al, WINA_ATTRIBUTES
13234 0000407B AA                      <1>       stosb  ; WinAAttributes
13235 0000407C 30C0                    <1>       xor    al, al ; WinBAttributes = 0
13236 0000407E AA                      <1>       stosb
13237 0000407F 66B84000                <1>       mov    ax, VBE_DISPI_BANK_SIZE_KB
13238 00004083 66AB                    <1>       stosw  ; WinGranularity
13239 00004085 66AB                    <1>       stosw  ; WinSize
13240 00004087 66B800A0                <1>       mov    ax, VGAMEM_GRAPH
13241 0000408B 66AB                    <1>       stosw  ; WinASegment
13242 0000408D 29C0                    <1>       sub    eax, eax
13243 0000408F 66AB                    <1>       stosw  ; WinBSegment = 0
13244 00004091 AB                      <1>       stosd  ; WinFuncPtr = 0
13245                                  <1>
13246 00004092 58                      <1>       pop    eax ; * ; pitch
13247 00004093 89C3                    <1>       mov    ebx, eax  ; high word of ebx = 0 ; 14/12/2020
13248 00004095 66AB                    <1>       stosw  ; BytesPerScanLine
13249                                  <1>
13250 00004097 5A                      <1>       pop    edx ; ** ; depth (bits per pixel)
13251 00004098 58                      <1>       pop    eax ; *** width, height
13252                                  <1>
13253                                  <1>       ; // Mandatory information for VBE 1.2 and above
13254                                  <1>
13255 00004099 66AB                    <1>       stosw  ; XResolution (width)
13256 0000409B C1E810                  <1>       shr    eax, 16
13257 0000409E 50                      <1>       push   eax ; **** height
13258 0000409F 66AB                    <1>       stosw  ; YResolution (height)
13259 000040A1 B008                    <1>       mov    al, 8
13260 000040A3 AA                      <1>       stosb  ; XCharSize  ; char width
13261 000040A4 B010                    <1>       mov    al, 16
13262 000040A6 AA                      <1>       stosb  ; YCharSize  ; char height
13263 000040A7 B001                    <1>       mov    al, 1
13264 000040A9 AA                      <1>       stosb  ; NumberOfPlanes
13265                                  <1>       ;movzx eax, dl
13266 000040AA 88D0                    <1>       mov    al, dl ; eax <= 32
13267 000040AC AA                      <1>       stosb  ; BitsPerPixel
13268                                  <1>       ; Number of banks = (height * pitch + 65535) / 65536
13269 000040AD 58                      <1>       pop    eax ; **** ; height
13270                                  <1>       ; 14/12/2020
13271 000040AE 52                      <1>       push   edx ; ***** ; depth ; edx <= 32
13272 000040AF F7E3                    <1>       mul    ebx  ; pitch (ebx) * height (eax)
13273                                  <1>       ;mov   edx, [esp] ; *****
13274                                  <1>       ;mov   dl, [esp] ; *****
13275 000040B1 05FFFF0000              <1>       add    eax, 65535
13276 000040B6 C1E810                  <1>       shr    eax, 16 ; / 65536 ; <= 127 ; 14/12/2020
13277 000040B9 AA                      <1>       stosb  ; NumberOfBanks
13278                                  <1>       ; 14/12/2020
13279                                  <1>       ;cmp   dl, 8 ; 8 bits per pixel
13280 000040BA 803C2408                <1>       cmp    byte [esp], 8
13281 000040BE 7704                    <1>       ja     short sml_2
13282 000040C0 B004                    <1>       mov    al, VBE_MEMORYMODEL_PACKED_PIXEL
13283 000040C2 EB02                    <1>       jmp    short sml_3
```

```
13284                                   <1> sml_2:
13285                                   <1>       ; 16, 24, 32 bits per pixel
13286 000040C4 B006                     <1>       mov   al, VBE_MEMORYMODEL_DIRECT_COLOR
13287                                   <1> sml_3:
13288 000040C6 AA                       <1>       stosb
13289 000040C7 30C0                     <1>       xor   al, al ; 0
13290 000040C9 AA                       <1>       stosb ; BankSize = 0
13291 000040CA 49                       <1>       dec   ecx ; pages - 1
13292                                   <1>       ; NumberOfImagePages = 262 for 320x200x8 mode
13293                                   <1>       ;;mov  ax, 255
13294                                   <1>       ; 14/12/2020
13295                                   <1>       ;mov   al, 255
13296 000040CB FEC8                     <1>       dec   al ; 255
13297 000040CD 39C1                     <1>       cmp   ecx, eax ; ecx <= 261, eax = 255
13298                                   <1>       ;cmp   cx, ax
13299 000040CF 7302                     <1>       jnb   short sml_4
13300 000040D1 88C8                     <1>       mov   al, cl
13301                                   <1> sml_4:
13302 000040D3 AA                       <1>       stosb ; NumberOfImagePages (1 byte)
13303 000040D4 28C0                     <1>       sub   al, al
13304 000040D6 AA                       <1>       stosb ; Reserved_page = 0
13305 000040D7 58                       <1>       pop   eax ; ***** ; depth
13306 000040D8 88C1                     <1>       mov   cl, al
13307                                   <1>       ; eax < = 32
13308 000040DA 2C08                     <1>       sub   al, 8 ; 8->0, 16->8, 24->16, 32->24
13309 000040DC BE[E6720000]             <1>       mov   esi, direct_color_fields
13310 000040E1 01C6                     <1>       add   esi, eax
13311 000040E3 56                       <1>       push  esi ; ******
13312 000040E4 AD                       <1>       lodsd ; RedMaskSize (AL), RedFieldPosition (AH)
13313                                   <1>             ; GreenMaskSize (16), GreenFieldPosition (24)
13314 000040E5 AB                       <1>       stosd
13315 000040E6 AD                       <1>       lodsd ; BlueMaskSize (AL), BlueFieldPosition (AH)
13316                                   <1>             ; RsvdMaskSize (16), RsvdFieldPosition (24)
13317 000040E7 AB                       <1>       stosd
13318 000040E8 5E                       <1>       pop   esi ; ******
13319                                   <1>
13320 000040E9 30C0                     <1>       xor   al, al ; 0
13321 000040EB 80F920                   <1>       cmp   cl, 32
13322 000040EE 7202                     <1>       jb    short sml_5
13323 000040F0 B002                     <1>       mov   al, VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
13324                                   <1> sml_5:
13325 000040F2 AA                       <1>       stosb ; DirectColorModeInfo
13326                                   <1>
13327                                   <1>       ; // Mandatory information for VBE 2.0 and above
13328                                   <1>
13329 000040F3 B8000000E0               <1>       mov   eax, VBE_DISPI_LFB_PHYSICAL_ADDRESS
13330 000040F8 AB                       <1>       stosd ; PhysBasePtr
13331 000040F9 29C0                     <1>       sub   eax, eax
13332 000040FB AB                       <1>       stosd ; OffScreenMemOffset = 0
13333 000040FC 66AB                     <1>       stosw ; OffScreenMemSize = 0
13334                                   <1>
13335                                   <1>       ;// Mandatory information for VBE 3.0 and above
13336                                   <1>
13337                                   <1>       ; ebx = pitch
13338 000040FE 6689D8                  <1>       mov   ax, bx
13339                                   <1>       ;stosw
13340                                   <1>
13341                                   <1>       ;xor   al, al
13342                                   <1>       ;stosb ; BnkNumberOfPages = 0
13343                                   <1>       ;stosb ; LinNumberOfPages = 0
13344                                   <1>
13345 00004101 AB                       <1>       stosd ; pitch (word), 0 (byte), 0 (byte)
13346                                   <1>
13347 00004102 AD                       <1>       lodsd ; LinRedMaskSize (AL), LinRedFieldPosition (AH)
13348                                   <1>             ; LinGreenMaskSize (16), LinGreenFieldPosition (24)
13349 00004103 AB                       <1>       stosd
13350 00004104 AD                       <1>       lodsd ; LinBlueMaskSize (AL), LinBlueFieldPosition (AH)
13351                                   <1>             ; LinRsvdMaskSize (16), LinRsvdFieldPosition (24)
13352 00004105 AB                       <1>       stosd
13353                                   <1>
13354 00004106 29C0                     <1>       sub   eax, eax
13355 00004108 AB                       <1>       stosd ; MaxPixelClock = 0
13356                                   <1>
13357                                   <1>       ;mov   eax, MODE_INFO_LIST
13358                                   <1>       ; 11/12/2020
13359 00004109 BE[3A120300]             <1>       mov   esi, MODE_INFO_LIST
13360                                   <1>
13361 0000410E C3                       <1>       retn
13362                                   <1>
13363                                   <1> ; end of set_mode_info_list ; edi = set_mode_info_list + 68
13364                                   <1>
13365                                   <1> pci_get_lfb_addr:
13366                                   <1>       ; 11/12/2020
13367                                   <1>       ; Get linear frame buffer base from PCI
13368                                   <1>       ; (vgabios.c, 02/01/2020, vruppert)
13369                                   <1>       ;
13370                                   <1>       ; Input:
13371                                   <1>       ;     ax = PCI device vendor id
13372                                   <1>       ; Output:
13373                                   <1>       ;     ax  = LFB address (high 16 bit) (zf=0)
13374                                   <1>       ;     eax = 0 -> not found (error) (zf=1)
13375                                   <1>       ;
13376                                   <1>       ; Modified registers: eax
13377                                   <1>
13378 0000410F 53                       <1>       push  ebx
13379 00004110 51                       <1>       push  ecx
13380 00004111 52                       <1>       push  edx
13381                                   <1>       ;
13382 00004112 89C3                     <1>       mov   ebx, eax
13383 00004114 31C9                     <1>       xor   ecx, ecx
13384 00004116 28D2                     <1>       sub   dl, dl ; mov dl, 0
13385 00004118 E842000000               <1>       call  pci_read_reg
13386 0000411D 6683F8FF                 <1>       cmp   ax, 0FFFFh
13387 00004121 7417                     <1>       je    short pci_get_lfb_addr_fail
13388                                   <1> pci_get_lfb_addr_next_dev:
```

```
13389 00004123 28D2              <1>       sub    dl, dl ; mov dl, 0
13390 00004125 E835000000        <1>       call   pci_read_reg
13391 0000412A 6639D8            <1>       cmp    ax, bx ; check vendor
13392 0000412D 740F              <1>       je     short pci_get_lfb_addr_found
13393 0000412F 6683C108          <1>       add    cx, 08h
13394 00004133 6681F90002        <1>       cmp    cx, 200h ; search bus 0 and 1
13395 00004138 72E9              <1>       jb     short pci_get_lfb_addr_next_dev
13396                            <1> pci_get_lfb_addr_fail:
13397 0000413A 31C0              <1>       xor    eax, eax ; no LFB
13398                            <1>       ; zf = 1
13399 0000413C EB1D              <1>       jmp    short pci_get_lfb_addr_return
13400                            <1> pci_get_lfb_addr_found:
13401 0000413E B210              <1>       mov    dl, 10h      ; I/O space 0
13402 00004140 E81A000000        <1>       call   pci_read_reg
13403 00004145 66A9F1FF          <1>       test   ax, 0FFF1h
13404 00004149 740D              <1>       jz     short pci_get_lfb_addr_success
13405 0000414B B214              <1>       mov    dl, 14h ; I/O space 1
13406 0000414D E80D000000        <1>       call   pci_read_reg
13407 00004152 66A9F1FF          <1>       test   ax, 0FFF1h
13408 00004156 75E2              <1>       jnz    short pci_get_lfb_addr_fail
13409                            <1> pci_get_lfb_addr_success:
13410 00004158 C1E810            <1>       shr    eax, 16 ; LFB address (hw)
13411                            <1>       ; zf = 0
13412                            <1> pci_get_lfb_addr_return:
13413 0000415B 5A                <1>       pop    edx
13414 0000415C 59                <1>       pop    ecx
13415 0000415D 5B                <1>       pop    ebx
13416 0000415E C3                <1>       retn
13417                            <1>
13418                            <1> pci_read_reg:
13419                            <1>       ; 11/12/2020
13420                            <1>       ; Read PCI register
13421                            <1>       ; (vgabios.c, 02/01/2020, vruppert)
13422                            <1>       ;
13423                            <1>       ; Input:
13424                            <1>       ;      cx = device/function
13425                            <1>       ;      dl = register
13426                            <1>       ; Output:
13427                            <1>       ;      eax = value
13428                            <1>       ;
13429                            <1>       ; Modified registers: eax, edx
13430                            <1>
13431 0000415F B800008000        <1>       mov    eax, 00800000h
13432 00004164 6689C8            <1>       mov    ax, cx
13433 00004167 C1E008            <1>       shl    eax, 8
13434 0000416A 88D0              <1>       mov    al, dl
13435 0000416C 66BAF80C          <1>       mov    dx, 0CF8h
13436 00004170 EF                <1>       out    dx, eax
13437 00004171 80C204            <1>       add    dl, 4 ; mov dx, 0CFCh
13438 00004174 ED                <1>       in     eax, dx
13439 00004175 C3                <1>       retn
13440                            <1>
13441                            <1> %endif
13442                            <1>
13443                            <1> ; -----------
13444                            <1>
13445                            <1> %if 0
13446                            <1>
13447                            <1> mode_info_find_mode:
13448                            <1>       ; 25/11/2020
13449                            <1>       ; Input:
13450                            <1>       ;      bx = VESA VBE2 video mode (+ bochs extensions)
13451                            <1>       ; Output:
13452                            <1>       ;      esi = mode info address (for BX input)
13453                            <1>       ;      esi = 0 -> not found
13454                            <1>       ;
13455                            <1>       ; Modified registers: eax, esi
13456                            <1>
13457                            <1>       xor    eax, eax
13458                            <1>       mov    esi, MODE_INFO_LIST
13459                            <1> mifm_1:
13460                            <1>       mov    ax, [esi]
13461                            <1>       cmp    ax, bx
13462                            <1>       je     short mifm_2
13463                            <1>       add    esi, MODEINFO.size ; add esi, 68
13464                            <1>       cmp    esi, VBE_VESA_MODE_END_OF_LIST
13465                            <1>       jb     short mifm_1
13466                            <1>       ; not found
13467                            <1>       sub    esi, esi ; 0
13468                            <1> mifm_2
13469                            <1>       retn
13470                            <1>
13471                            <1> dispi_get_bpp:
13472                            <1>       ; 28/11/2020
13473                            <1>       ; Input:
13474                            <1>       ;      none
13475                            <1>       ; Output:
13476                            <1>       ;      al = Bits per pixel
13477                            <1>       ;           (8,16,24,32)
13478                            <1>       ;      ah = Bytes per pixel
13479                            <1>       ;           (1,2,3,4)
13480                            <1>       ;
13481                            <1>       ; Modified registers: none
13482                            <1>
13483                            <1>       ;push  edx
13484                            <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
13485                            <1>       ;mov   ax, 03h        ; VBE_DISPI_INDEX_BPP
13486                            <1>       ;out   dx, ax
13487                            <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
13488                            <1>       ;;mov  dl, 0CFh
13489                            <1>       ;inc   dl
13490                            <1>       ;in    ax, dx
13491                            <1>       ;mov   ah, al
13492                            <1>       ;shr   ah, 3 ; / 8
13493                            <1>       ;;test al, 7 ; 15 bit graphics mode
```

```
13494                           <1>       ;;jz   short get_bpp_noinc
13495                           <1>       ;;inc  ah
13496                           <1> ;;get_bpp_noinc:
13497                           <1>       ;pop   edx
13498                           <1>       ;retn
13499                           <1>
13500                           <1>       ; 28/11/2020
13501                           <1>       ; Modified registers: edx
13502                           <1>       ;push  edx
13503                           <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
13504                           <1>       call   dispi_get_parms
13505                           <1>       ;pop   edx
13506                           <1>       ;retn
13507                           <1>       mov    ah, al
13508                           <1>       shr    ah, 3 ; / 8
13509                           <1>       ;test  al, 7 ; 15 bit graphips mode
13510                           <1>       ;jz    short get_bpp_noinc
13511                           <1>       ;inc   ah
13512                           <1> ;get_bpp_noinc:
13513                           <1>       ;pop   edx
13514                           <1>       retn
13515                           <1>
13516                           <1> restore_vesa_video_state:
13517                           <1>       ; 14/01/2021
13518                           <1>       ; 06/12/2020
13519                           <1>       ; Input:
13520                           <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
13521                           <1>       ; Output:
13522                           <1>       ;      AX = 004Fh (successed)
13523                           <1>       ;
13524                           <1>       ;      eax = 0 -> buffer size problem
13525                           <1>       ;      eax > 0 and ax <> 004Fh -> failed
13526                           <1>       ;
13527                           <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
13528                           <1>
13529                           <1>       ;movzx ecx, word [vbe3stbufsize]
13530                           <1>       ;cmp   cx, 32 ; 32 * 64 bytes
13531                           <1>       ;ja    short r_v_b_s_fail
13532                           <1>
13533                           <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
13534                           <1>       shl    cx, 4 ; dword count for movsd
13535                           <1>       mov    edi, VBE3SAVERESTOREBLOCK ; destination
13536                           <1>                                ; (vbe3 pmi buff)
13537                           <1>       mov    esi, VBE3VIDEOSTATE ; source (kernel buff)
13538                           <1>       rep    movsd
13539                           <1>
13540                           <1>       mov    ax, 4F04h
13541                           <1>       mov    dl, 02h ; restore
13542                           <1>       ;mov   cx, 0Fh
13543                           <1>       mov    cl, 0Fh
13544                           <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
13545                           <1>       jmp    short int10h_32bit_pmi
13546                           <1>
13547                           <1> ;s_v_b_s_fail:
13548                           <1> ;r_v_b_s_fail:
13549                           <1> ;     xor    eax, eax
13550                           <1> ;     retn
13551                           <1>
13552                           <1> save_vesa_video_state:
13553                           <1>       ; 14/01/2021
13554                           <1>       ; 06/12/2020
13555                           <1>       ; Input:
13556                           <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
13557                           <1>       ; Output:
13558                           <1>       ;      AX = 004Fh (successed)
13559                           <1>       ;
13560                           <1>       ;      eax = 0 -> buffer size problem
13561                           <1>       ;      eax > 0 and ax <> 004Fh -> failed
13562                           <1>       ;
13563                           <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
13564                           <1>
13565                           <1>       ;cmp   word [vbe33stbufsize], 32
13566                           <1>       ;                  ; 32 * 64 bytes
13567                           <1>       ;ja    short s_v_b_s_fail
13568                           <1>
13569                           <1>       mov    ax, 4F04h
13570                           <1>       mov    dl, 01h ; save
13571                           <1>       ;mov   cx, 0Fh
13572                           <1>       mov    cl, 0Fh
13573                           <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
13574                           <1>
13575                           <1>       call   int10h_32bit_pmi
13576                           <1>
13577                           <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
13578                           <1>       shl    cx, 4 ; dword count for movsd
13579                           <1>       mov    esi, VBE3SAVERESTOREBLOCK ; destination
13580                           <1>                                ; (vbe3 pmi buff)
13581                           <1>       mov    edi, VBE3VIDEOSTATE ; source (kernel buff)
13582                           <1>       rep    movsd
13583                           <1>       retn
13584                           <1>
13585                           <1> dispi_set_bank_farcall:
13586                           <1>       ; 12/04/2021 (32 bit push/pop)
13587                           <1>       ; 11/12/2020
13588                           <1>       ; (This may be 'sysvideo' function, later)
13589                           <1>       ;
13590                           <1>       ; Input:
13591                           <1>       ;      bx = 0000h, set bank number
13592                           <1>       ;         = 0100h, get bank number
13593                           <1>       ;      dx = bank number (if bx = 0)
13594                           <1>       ; Output:
13595                           <1>       ;      dx = bank number
13596                           <1>
13597                           <1>       cmp    bx, 0100h
13598                           <1>       je     short dispi_set_bank_farcall_get
```

```
13599                                  <1>       or    bx, bx
13600                                  <1>       jnz   dispi_set_bank_farcall_error
13601                                  <1>       mov   ax, dx
13602                                  <1>       ;push  dx
13603                                  <1>       ;push  ax
13604                                  <1>       ; 12/04/2021
13605                                  <1>       push  edx
13606                                  <1>       push  eax
13607                                  <1>       mov   ax, VBE_DISPI_INDEX_BANK
13608                                  <1>       mov   dx, VBE_DISPI_IOPORT_INDEX
13609                                  <1>       out   dx, ax
13610                                  <1>       ;pop   ax
13611                                  <1>       ; 12/04/2021
13612                                  <1>       pop   eax
13613                                  <1>       mov   dx, VBE_DISPI_IOPORT_DATA
13614                                  <1>       out   dx, ax
13615                                  <1>       in    ax, dx
13616                                  <1>       ;pop   dx
13617                                  <1>       ; 12/04/2021
13618                                  <1>       pop   edx
13619                                  <1>       cmp   dx, ax
13620                                  <1>       jne   short dispi_set_bank_farcall_error
13621                                  <1>       mov   ax, 004Fh
13622                                  <1>       retn  ; retf for real mode far call
13623                                  <1> dispi_set_bank_farcall_get:
13624                                  <1>       mov   ax, VBE_DISPI_INDEX_BANK
13625                                  <1>       mov   dx, VBE_DISPI_IOPORT_INDEX
13626                                  <1>       out   dx, ax
13627                                  <1>       mov   dx, VBE_DISPI_IOPORT_DATA
13628                                  <1>       in    ax, dx
13629                                  <1>       mov   dx, ax
13630                                  <1>       retn  ; retf for real mode far call
13631                                  <1> dispi_set_bank_farcall_error:
13632                                  <1>       mov   ax, 014Fh
13633                                  <1>       retn  ; retf for real mode far call
13634                                  <1>
13635                                  <1> %endif
13636                                  <1>
13637                                  <1> ; % include 'vidata.s' ; VIDEO DATA
13638                                  <1>
13639                                  <1> ; /// End Of VIDEO FUNCTIONS ///
 2660
 2661                                      setup_rtc_int:
 2662                                      ; source: http://wiki.osdev.org/RTC
 2663 00004176 FA                           cli          ; disable interrupts
 2664                                       ; default int frequency is 1024 Hz (Lower 4 bits of register A is 0110b or 6)
 2665                                       ; in order to change this ...
 2666                                       ; frequency  = 32768 >> (rate-1) --> 32768 >> 5 = 1024
 2667                                       ; (rate must be above 2 and not over 15)
 2668                                       ; new rate = 15 --> 32768 >> (15-1) = 2 Hz
 2669 00004177 B08A                        mov   al, 8Ah
 2670 00004179 E670                        out   70h, al ; set index to register A, disable NMI
 2671 0000417B 90                          nop
 2672 0000417C E471                        in    al, 71h ; get initial value of register A
 2673 0000417E 88C4                        mov   ah, al
 2674 00004180 80E4F0                      and   ah, 0F0h
 2675 00004183 B08A                        mov   al, 8Ah
 2676 00004185 E670                        out   70h, al ; reset index to register A
 2677 00004187 88E0                        mov   al, ah
 2678 00004189 0C0F                        or    al, 0Fh      ; new rate (0Fh -> 15)
 2679 0000418B E671                        out   71h, al ; write only our rate to A. Note, rate is the bottom 4 bits.
 2680                                       ; enable RTC interrupt
 2681 0000418D B08B                        mov   al, 8Bh ;
 2682 0000418F E670                        out   70h, al ; select register B and disable NMI
 2683 00004191 90                          nop
 2684 00004192 E471                        in    al, 71h ; read the current value of register B
 2685 00004194 88C4                        mov   ah, al  ;
 2686 00004196 B08B                        mov   al, 8Bh ;
 2687 00004198 E670                        out   70h, al ; set the index again (a read will reset the index to register B)
 2688 0000419A 88E0                        mov   al, ah  ;
 2689 0000419C 0C40                        or    al, 40h ;
 2690 0000419E E671                        out   71h, al ; write the previous value ORed with 0x40. This turns on bit 6 of register B
 2691 000041A0 FB                          sti
 2692 000041A1 C3                          retn
 2693
 2694                                   ; Write memory information
 2695                                   ; 29/01/2016
 2696                                   ; 06/11/2014
 2697                                   ; 14/08/2015
 2698                                   memory_info:
 2699 000041A2 A1[F4880100]                mov   eax, [memory_size] ; in pages
 2700 000041A7 50                          push  eax
 2701 000041A8 C1E00C                      shl   eax, 12               ; in bytes
 2702 000041AB BB0A000000                  mov   ebx, 10
 2703 000041B0 89D9                        mov   ecx, ebx        ; 10
 2704 000041B2 BE[794B0100]                mov   esi, mem_total_b_str
 2705 000041B7 E8D7000000                  call  bintdstr
 2706 000041BC 58                          pop   eax
 2707 000041BD B107                        mov   cl, 7
 2708 000041BF BE[9D4B0100]                mov   esi, mem_total_p_str
 2709 000041C4 E8CA000000                  call  bintdstr
 2710                                       ; 14/08/2015
 2711 000041C9 E8E2000000                  call  calc_free_mem
 2712                                       ; edx = calculated free pages
 2713                                       ; ecx = 0
 2714 000041CE A1[F8880100]                mov   eax, [free_pages]
 2715 000041D3 39D0                        cmp   eax, edx ; calculated free mem value
 2716                                             ; and initial free mem value are same or not?
 2717 000041D5 751D                        jne   short pmim ; print mem info with '?' if not
 2718 000041D7 52                          push  edx ; free memory in pages
 2719                                       ;mov  eax, edx
 2720 000041D8 C1E00C                      shl   eax, 12 ; convert page count
 2721                                                   ; to byte count
 2722 000041DB B10A                        mov   cl, 10
 2723 000041DD BE[BD4B0100]                mov   esi, free_mem_b_str
```

```
2724 000041E2 E8AC000000                call  bintdstr
2725 000041E7 58                        pop   eax
2726 000041E8 B107                      mov   cl, 7
2727 000041EA BE[E14B0100]              mov   esi, free_mem_p_str
2728 000041EF E89F000000                call  bintdstr
2729                                pmim:
2730 000041F4 BE[674B0100]              mov   esi, msg_memory_info
2731                                    ;
2732 000041F9 B407                      mov   ah, 07h ; Black background,
2733                                                  ; light gray forecolor
2734                                print_kmsg: ; 29/01/2016
2735 000041FB 8825[1F890100]            mov   [ccolor], ah
2736                                pkmsg_loop:
2737 00004201 AC                        lodsb
2738 00004202 08C0                      or    al, al
2739 00004204 7410                      jz    short pkmsg_ok
2740 00004206 56                        push  esi
2741                                    ; 13/05/2016
2742 00004207 0FB61D[1F890100]          movzx ebx, byte [ccolor]
2743                                                  ; Video page 0 (bh=0)
2744 0000420E E86FE0FFFF                call  _write_tty
2745 00004213 5E                        pop   esi
2746 00004214 EBEB                      jmp   short pkmsg_loop
2747                                pkmsg_ok:
2748 00004216 C3                        retn
2749
2750                                ; 19/12/2020
2751                                ; temporary
2752                                ; Write default liner frame buffer address
2753                                ;
2754                                default_lfb_info:
2755 00004217 66A1[E10E0000]            mov   ax, [def_LFB_addr] ; high word
2756 0000421D E829000000                call  wordtohex
2757 00004222 A3[394C0100]              mov   dword [lfb_addr_str], eax
2758 00004227 BE[224C0100]              mov   esi, msg_lfb_addr
2759 0000422C B40F                      mov   ah, 0Fh ; Black background,
2760                                                  ; white forecolor
2761 0000422E EBCB                      jmp   short print_kmsg
2762
2763                                ; Convert binary number to hexadecimal string
2764                                ; 10/05/2015
2765                                ; dsectpm.s (28/02/2015)
2766                                ; Retro UNIX 386 v1 - Kernel v0.2.0.6
2767                                ; 01/12/2014
2768                                ; 25/11/2014
2769                                ;
2770                                bytetohex:
2771                                    ; INPUT ->
2772                                    ;     AL = byte (binary number)
2773                                    ; OUTPUT ->
2774                                    ;     AX = hexadecimal string
2775                                    ;
2776 00004230 53                        push  ebx
2777 00004231 31DB                      xor   ebx, ebx
2778 00004233 88C3                      mov   bl, al
2779 00004235 C0EB04                    shr   bl, 4
2780 00004238 8A9B[82420000]            mov   bl, [ebx+hexchrs]
2781 0000423E 86D8                      xchg  bl, al
2782 00004240 80E30F                    and   bl, 0Fh
2783 00004243 8AA3[82420000]            mov   ah, [ebx+hexchrs]
2784 00004249 5B                        pop   ebx
2785 0000424A C3                        retn
2786
2787                                wordtohex:
2788                                    ; INPUT ->
2789                                    ;     AX = word (binary number)
2790                                    ; OUTPUT ->
2791                                    ;     EAX = hexadecimal string
2792                                    ;
2793 0000424B 53                        push  ebx
2794 0000424C 31DB                      xor   ebx, ebx
2795 0000424E 86E0                      xchg  ah, al
2796 00004250 6650                      push  ax
2797 00004252 88E3                      mov   bl, ah
2798 00004254 C0EB04                    shr   bl, 4
2799 00004257 8A83[82420000]            mov   al, [ebx+hexchrs]
2800 0000425D 88E3                      mov   bl, ah
2801 0000425F 80E30F                    and   bl, 0Fh
2802 00004262 8AA3[82420000]            mov   ah, [ebx+hexchrs]
2803 00004268 C1E010                    shl   eax, 16
2804 0000426B 6658                      pop   ax
2805 0000426D 5B                        pop   ebx
2806 0000426E EBC0                      jmp   short bytetohex
2807                                    ;mov  bl, al
2808                                    ;shr  bl, 4
2809                                    ;mov  bl, [ebx+hexchrs]
2810                                    ;xchg bl, al
2811                                    ;and  bl, 0Fh
2812                                    ;mov  ah, [ebx+hexchrs]
2813                                    ;pop  ebx
2814                                    ;retn
2815
2816                                dwordtohex:
2817                                    ; INPUT ->
2818                                    ;     EAX = dword (binary number)
2819                                    ; OUTPUT ->
2820                                    ;     EDX:EAX = hexadecimal string
2821                                    ;
2822 00004270 50                        push  eax
2823 00004271 C1E810                    shr   eax, 16
2824 00004274 E8D2FFFFFF                call  wordtohex
2825 00004279 89C2                      mov   edx, eax
2826 0000427B 58                        pop   eax
2827 0000427C E8CAFFFFFF                call  wordtohex
2828 00004281 C3                        retn
```

```
                                ; 10/05/2015
                                hex_digits:
                                hexchrs:
00004282 303132333435363738-            db '0123456789ABCDEF'
0000428B 39414243444546
                                ; 19/01/2021 - VESA EDID ready flag (4Fh)
00004292 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)
00004293 01CE                           add    esi, ecx
                                btdstr0:
00004295 4E                             dec    esi
00004296 31D2                           xor    edx, edx
00004298 F7F3                           div    ebx
0000429A 80C230                         add    dl, 30h
0000429D 8816                           mov    [esi], dl
0000429F FEC9                           dec    cl
000042A1 740C                           jz     short btdstr2 ; 08/09/2016
000042A3 09C0                           or     eax, eax
000042A5 75EE                           jnz    short btdstr0
                                btdstr1:
000042A7 4E                             dec    esi
000042A8 C60620                           mov     byte [esi], 20h ; blank space
000042AB FEC9                           dec    cl
000042AD 75F8                           jnz    short btdstr1
                                btdstr2:
000042AF C3                             retn

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

                                calc_free_mem:
000042B0 31D2                           xor    edx, edx
                                        ;xor   ecx, ecx
000042B2 668B0D[08890100]               mov    cx, [mat_size] ; in pages
000042B9 C1E10A                         shl    ecx, 10       ; 1024 dwords per page
000042BC BE00001000                     mov    esi, MEM_ALLOC_TBL
                                cfm0:
000042C1 AD                             lodsd
000042C2 51                             push   ecx
000042C3 B920000000                     mov    ecx, 32
                                cfm1:
000042C8 D1E8                           shr    eax, 1
000042CA 7301                           jnc    short cfm2
000042CC 42                             inc    edx
                                cfm2:
000042CD E2F9                           loop   cfm1
000042CF 59                             pop    ecx
000042D0 E2EF                           loop   cfm0
000042D2 C3                             retn

                                    %include 'diskio.s'  ; 07/03/2015
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - diskio.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 11/04/2021
                                <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
```

```
2933                            <1> ; 06/02/2015 (unix386.s)
2934                            <1> ; 16/12/2014 - 02/01/2015 (dsectrm2.s)
2935                            <1> ;
2936                            <1> ; Code (DELAY) modifications - AWARD BIOS 1999 (ADISK.EQU, COMMON.MAC)
2937                            <1> ;
2938                            <1> ; ADISK.EQU
2939                            <1>
2940                            <1> ;----- Wait control constants
2941                            <1>
2942                            <1> ;amount of time to wait while RESET is active.
2943                            <1>
2944                            <1> WAITCPU_RESET_ON   EQU    21          ;Reset on must last at least 14us
2945                            <1>                                       ;at 250 KBS xfer rate.
2946                            <1>                                       ;see INTEL MCS, 1985, pg. 5-456
2947                            <1>
2948                            <1> WAITCPU_FOR_STATUS EQU    100         ;allow 30 microseconds for
2949                            <1>                                       ;status register to become valid
2950                            <1>                                       ;before re-reading.
2951                            <1>
2952                            <1> ;After sending a byte to NEC, status register may remain
2953                            <1> ;incorrectly set for 24 us.
2954                            <1>
2955                            <1> WAITCPU_RQM_LOW          EQU    24          ;number of loops to check for
2956                            <1>                                       ;RQM low.
2957                            <1>
2958                            <1> ; COMMON.MAC
2959                            <1> ;
2960                            <1> ;     Timing macros
2961                            <1> ;
2962                            <1>
2963                            <1> %macro             SIODELAY 0               ; SHORT IODELAY
2964                            <1>             jmp short $+2
2965                            <1> %endmacro
2966                            <1>
2967                            <1> %macro             IODELAY  0               ; NORMAL IODELAY
2968                            <1>             jmp short $+2
2969                            <1>             jmp short $+2
2970                            <1> %endmacro
2971                            <1>
2972                            <1> %macro             NEWIODELAY 0
2973                            <1>             out    0ebh,al
2974                            <1> %endmacro
2975                            <1>
2976                            <1> ; (According to) AWARD BIOS 1999 - ATORGS.ASM (dw -> equ, db -> equ)
2977                            <1> ;;; WAIT_FOR_MEM
2978                            <1> ;WAIT_FDU_INT_LO   equ    017798       ; 2.5 secs in 30 micro units.
2979                            <1> ;WAIT_FDU_INT_HI   equ    1
2980                            <1> WAIT_FDU_INT_LH           equ    83334        ; 27/02/2015 (2.5 seconds waiting)
2981                            <1> ;;; WAIT_FOR_PORT
2982                            <1> ;WAIT_FDU_SEND_LO  equ    16667        ; .5 secons in 30 us units.
2983                            <1> ;WAIT_FDU_SEND_HI  equ    0
2984                            <1> WAIT_FDU_SEND_LH   equ    16667        ; 27/02/2015
2985                            <1> ;Time to wait while waiting for each byte of NEC results = .5
2986                            <1> ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
2987                            <1> ;WAIT_FDU_RESULTS_LO      equ    16667        ; .5 seconds in 30 micro units.
2988                            <1> ;WAIT_FDU_RESULTS_HI      equ    0
2989                            <1> WAIT_FDU_RESULTS_LH       equ    16667  ; 27/02/2015
2990                            <1> ;;; WAIT_REFRESH
2991                            <1> ;amount of time to wait for head settle, per unit in parameter
2992                            <1> ;table = 1 ms.
2993                            <1> WAIT_FDU_HEAD_SETTLE      equ    33           ; 1 ms in 30 micro units.
2994                            <1>
2995                            <1>
2996                            <1> ; ///////////////// DISKETTE I/O ////////////////
2997                            <1>
2998                            <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - POSTEQU.INC)
2999                            <1>
3000                            <1> ;--------------------------------------
3001                            <1> ;     EQUATES USED BY POST AND BIOS    :
3002                            <1> ;--------------------------------------
3003                            <1>
3004                            <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
3005                            <1> ;PORT_A           EQU    060H         ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
3006                            <1> ;PORT_B           EQU    061H         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
3007                            <1> ;REFRESH_BIT EQU    00010000B    ; REFRESH TEST BIT
3008                            <1>
3009                            <1> ;--------------------------------------
3010                            <1> ;     CMOS EQUATES FOR THIS SYSTEM     :
3011                            <1> ;-------------------------------------------------------------------------------
3012                            <1> ;CMOS_PORT   EQU    070H         ; I/O ADDRESS OF CMOS ADDRESS PORT
3013                            <1> ;CMOS_DATA   EQU    071H         ; I/O ADDRESS OF CMOS DATA PORT
3014                            <1> ;NMI         EQU    10000000B    ; DISABLE NMI INTERRUPTS MASK -
3015                            <1>                                 ;  HIGH BIT OF CMOS LOCATION ADDRESS
3016                            <1>
3017                            <1> ;---------- CMOS TABLE LOCATION ADDRESS'S ## ---------------------------------
3018                            <1> CMOS_DISKETTE      EQU    010H         ; DISKETTE DRIVE TYPE BYTE      ;
3019                            <1> ;            EQU    011H         ; - RESERVED                    ;C
3020                            <1> CMOS_DISK    EQU    012H         ; FIXED DISK TYPE BYTE                ;H
3021                            <1> ;            EQU    013H         ; - RESERVED                    ;E
3022                            <1> CMOS_EQUIP   EQU    014H         ; EQUIPMENT WORD LOW BYTE       ;C
3023                            <1>
3024                            <1> ;---------- DISKETTE EQUATES --------------------------------------------------
3025                            <1> INT_FLAG     EQU    10000000B    ; INTERRUPT OCCURRENCE FLAG
3026                            <1> DSK_CHG      EQU    10000000B    ; DISKETTE CHANGE FLAG MASK BIT
3027                            <1> DETERMINED   EQU    00010000B    ; SET STATE DETERMINED IN STATE BITS
3028                            <1> HOME         EQU    00010000B    ; TRACK 0 MASK
3029                            <1> SENSE_DRV_ST EQU    00000100B    ; SENSE DRIVE STATUS COMMAND
3030                            <1> TRK_SLAP     EQU    030H         ; CRASH STOP (48 TPI DRIVES)
3031                            <1> QUIET_SEEK   EQU    00AH         ; SEEK TO TRACK 10
3032                            <1> ;MAX_DRV     EQU    2            ; MAX NUMBER OF DRIVES
3033                            <1> HD12_SETTLE  EQU    15           ; 1.2 M HEAD SETTLE TIME
3034                            <1> HD320_SETTLE EQU    20           ; 320 K HEAD SETTLE TIME
3035                            <1> MOTOR_WAIT   EQU    37           ; 2 SECONDS OF COUNTS FOR MOTOR TURN OFF
3036                            <1>
3037                            <1> ;---------- DISKETTE ERRORS ---------------------------------------------------
```

```
3038                    <1> ;TIME_OUT    EQU    080H         ; ATTACHMENT FAILED TO RESPOND
3039                    <1> ;BAD_SEEK    EQU    040H         ; SEEK OPERATION FAILED
3040                    <1> BAD_NEC      EQU    020H         ; DISKETTE CONTROLLER HAS FAILED
3041                    <1> BAD_CRC      EQU    010H         ; BAD CRC ON DISKETTE READ
3042                    <1> MED_NOT_FND  EQU    00CH         ; MEDIA TYPE NOT FOUND
3043                    <1> DMA_BOUNDARY EQU    009H         ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
3044                    <1> BAD_DMA      EQU    008H         ; DMA OVERRUN ON OPERATION
3045                    <1> MEDIA_CHANGE EQU    006H         ; MEDIA REMOVED ON DUAL ATTACH CARD
3046                    <1> RECORD_NOT_FND      EQU    004H         ; REQUESTED SECTOR NOT FOUND
3047                    <1> WRITE_PROTECT       EQU    003H         ; WRITE ATTEMPTED ON WRITE PROTECT DISK
3048                    <1> BAD_ADDR_MARK       EQU    002H         ; ADDRESS MARK NOT FOUND
3049                    <1> BAD_CMD      EQU    001H         ; BAD COMMAND PASSED TO DISKETTE I/O
3050                    <1>
3051                    <1> ;---------- DISK CHANGE LINE EQUATES -----------------------------------------
3052                    <1> NOCHGLN      EQU    001H         ; NO DISK CHANGE LINE AVAILABLE
3053                    <1> CHGLN        EQU    002H         ; DISK CHANGE LINE AVAILABLE
3054                    <1>
3055                    <1> ;---------- MEDIA/DRIVE STATE INDICATORS -------------------------------------
3056                    <1> TRK_CAPA     EQU    00000001B    ; 80 TRACK CAPABILITY
3057                    <1> FMT_CAPA     EQU    00000010B    ; MULTIPLE FORMAT CAPABILITY (1.2M)
3058                    <1> DRV_DET      EQU    00000100B    ; DRIVE DETERMINED
3059                    <1> MED_DET      EQU    00010000B    ; MEDIA DETERMINED BIT
3060                    <1> DBL_STEP     EQU    00100000B    ; DOUBLE STEP BIT
3061                    <1> RATE_MSK     EQU    11000000B    ; MASK FOR CLEARING ALL BUT RATE
3062                    <1> RATE_500     EQU    00000000B    ; 500 KBS DATA RATE
3063                    <1> RATE_300     EQU    01000000B    ; 300 KBS DATA RATE
3064                    <1> RATE_250     EQU    10000000B    ; 250 KBS DATA RATE
3065                    <1> STRT_MSK     EQU    00001100B    ; OPERATION START RATE MASK
3066                    <1> SEND_MSK     EQU    11000000B    ; MASK FOR SEND RATE BITS
3067                    <1>
3068                    <1> ;---------- MEDIA/DRIVE STATE INDICATORS COMPATIBILITY ------------------------
3069                    <1> M3D3U        EQU    00000000B    ; 360 MEDIA/DRIVE NOT ESTABLISHED
3070                    <1> M3D1U        EQU    00000001B    ; 360 MEDIA,1.2DRIVE NOT ESTABLISHED
3071                    <1> M1D1U        EQU    00000010B    ; 1.2 MEDIA/DRIVE NOT ESTABLISHED
3072                    <1> MED_UNK      EQU    00000111B    ; NONE OF THE ABOVE
3073                    <1>
3074                    <1> ;---------- INTERRUPT EQUATES ------------------------------------------------
3075                    <1> ;EOI         EQU    020H         ; END OF INTERRUPT COMMAND TO 8259
3076                    <1> ;INTA00             EQU    020H         ; 8259 PORT
3077                    <1> INTA01              EQU    021H         ; 8259 PORT
3078                    <1> INTB00              EQU    0A0H         ; 2ND 8259
3079                    <1> INTB01              EQU    0A1H         ;
3080                    <1>
3081                    <1> ;--------------------------------------------------------------------------------
3082                    <1> DMA08        EQU    008H         ; DMA STATUS REGISTER PORT ADDRESS
3083                    <1> DMA          EQU    000H         ; DMA CH.0 ADDRESS REGISTER PORT ADDRESS
3084                    <1> DMA18        EQU    0D0H         ; 2ND DMA STATUS PORT ADDRESS
3085                    <1> DMA1         EQU    0C0H         ; 2ND DMA CH.0 ADDRESS REGISTER ADDRESS
3086                    <1> ;--------------------------------------------------------------------------------
3087                    <1> ;TIMER              EQU    040H         ; 8254 TIMER - BASE ADDRESS
3088                    <1>
3089                    <1> ;--------------------------------------------------------------------------------
3090                    <1> DMA_PAGE     EQU    081H         ; START OF DMA PAGE REGISTERS
3091                    <1>
3092                    <1> ; 06/02/2015 (unix386.s, protected mode modifications)
3093                    <1> ; (unix386.s <-- dsectrm2.s)
3094                    <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - DSEG.INC)
3095                    <1>
3096                    <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
3097                    <1> ; 10/12/2014
3098                    <1> ;
3099                    <1> ;int40h:
3100                    <1> ;    pushf
3101                    <1> ;    push   cs
3102                    <1> ;    ;cli
3103                    <1> ;    call   DISKETTE_IO_1
3104                    <1> ;    retn
3105                    <1>
3106                    <1> ; DSKETTE ----- 04/21/86 DISKETTE BIOS
3107                    <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
3108                    <1> ;
3109                    <1>
3110                    <1> ;-- INT13H ---------------------------------------------------------------------
3111                    <1> ; DISKETTE I/O
3112                    <1> ;    THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 INCH 360 KB,
3113                    <1> ;    1.2 MB, 720 KB AND 1.44 MB DISKETTE DRIVES.
3114                    <1> ; INPUT
3115                    <1> ;    (AH) =  00H RESET DISKETTE SYSTEM
3116                    <1> ;            HARD RESET TO NEC, PREPARE COMMAND, RECALIBRATE REQUIRED
3117                    <1> ;            ON ALL DRIVES
3118                    <1> ;--------------------------------------------------------------------------------
3119                    <1> ;    (AH)= 01H  READ THE STATUS OF THE SYSTEM INTO (AH)
3120                    <1> ;            @DISKETTE_STATUS FROM LAST OPERATION IS USED
3121                    <1> ;--------------------------------------------------------------------------------
3122                    <1> ;    REGISTERS FOR READ/WRITE/VERIFY/FORMAT
3123                    <1> ;    (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3124                    <1> ;    (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
3125                    <1> ;    (CH) - TRACK NUMBER (NOT VALUE CHECKED)
3126                    <1> ;           MEDIA  DRIVE  TRACK NUMBER
3127                    <1> ;           320/360      320/360          0-39
3128                    <1> ;           320/360      1.2M       0-39
3129                    <1> ;           1.2M   1.2M       0-79
3130                    <1> ;           720K   720K       0-79
3131                    <1> ;           1.44M  1.44M      0-79
3132                    <1> ;    (CL) -       SECTOR NUMBER (NOT VALUE CHECKED, NOT USED FOR FORMAT)
3133                    <1> ;           MEDIA  DRIVE  SECTOR NUMBER
3134                    <1> ;           320/360      320/360           1-8/9
3135                    <1> ;           320/360      1.2M        1-8/9
3136                    <1> ;           1.2M   1.2M       1-15
3137                    <1> ;           720K   720K       1-9
3138                    <1> ;           1.44M  1.44M      1-18
3139                    <1> ;    (AL)   NUMBER OF SECTORS (NOT VALUE CHECKED)
3140                    <1> ;           MEDIA  DRIVE  MAX NUMBER OF SECTORS
3141                    <1> ;           320/360      320/360              8/9
3142                    <1> ;           320/360      1.2M          8/9
```

```
3143                    <1> ;           1.2M   1.2M         15
3144                    <1> ;           720K   720K         9
3145                    <1> ;           1.44M  1.44M        18
3146                    <1> ;
3147                    <1> ;     (ES:BX) - ADDRESS OF BUFFER (NOT REQUIRED FOR VERIFY)
3148                    <1> ;
3149                    <1> ;---------------------------------------------------------------------------
3150                    <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY
3151                    <1> ;---------------------------------------------------------------------------
3152                    <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY
3153                    <1> ;---------------------------------------------------------------------------
3154                    <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS
3155                    <1> ;---------------------------------------------------------------------------
3156                    <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK
3157                    <1> ;           (ES,BX) MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS
3158                    <1> ;           FOR THE      TRACK. EACH FIELD IS COMPOSED OF 4 BYTES, (C,H,R,N),
3159                    <1> ;           WHERE C = TRACK NUMBER, H=HEAD NUMBER, R = SECTOR NUMBER,
3160                    <1> ;           N= NUMBER OF BYTES PER SECTOR (00=128,01=256,02=512,03=1024),
3161                    <1> ;           THERE MUST BE ONE ENTRY FOR EVERY SECTOR ON THE TRACK.
3162                    <1> ;           THIS INFORMATION IS USED TO FIND THE REQUESTED SECTOR DURING
3163                    <1> ;           READ/WRITE ACCESS.
3164                    <1> ;           PRIOR TO FORMATTING A DISKETTE, IF THERE EXISTS MORE THAN
3165                    <1> ;           ONE SUPPORTED MEDIA FORMAT TYPE WITHIN THE DRIVE IN QUESTION,
3166                    <1> ;           THEN "SET DASD TYPE" (INT 13H, AH = 17H) OR 'SET MEDIA TYPE'
3167                    <1> ;           (INT 13H, AH =  18H) MUST BE CALLED TO SET THE DISKETTE TYPE
3168                    <1> ;           THAT IS TO BE FORMATTED. IF "SET DASD TYPE" OR "SET MEDIA TYPE"
3169                    <1> ;           IS NOT CALLED, THE FORMAT ROUTINE WILL ASSUME THE
3170                    <1> ;           MEDIA FORMAT TO BE THE MAXIMUM CAPACITY OF THE DRIVE.
3171                    <1> ;
3172                    <1> ;           THESE PARAMETERS OF DISK BASE MUST BE CHANGED IN ORDER TO
3173                    <1> ;           FORMAT THE FOLLOWING MEDIAS:
3174                    <1> ;           ------------------------------------------
3175                    <1> ;           : MEDIA  :     DRIVE      : PARM 1 : PARM 2 :
3176                    <1> ;           ------------------------------------------
3177                    <1> ;           : 320K  : 320K/360K/1.2M :  50H   :   8    :
3178                    <1> ;           : 360K  : 320K/360K/1.2M :  50H   :   9    :
3179                    <1> ;           : 1.2M  : 1.2M           :  54H   :  15    :
3180                    <1> ;           : 720K  : 720K/1.44M     :  50H   :   9    :
3181                    <1> ;           : 1.44M       : 1.44M          :  6CH   :  18    :
3182                    <1> ;           ------------------------------------------
3183                    <1> ;           NOTES: - PARM 1 = GAP LENGTH FOR FORMAT
3184                    <1> ;                  - PARM 2 = EOT (LAST SECTOR ON TRACK)
3185                    <1> ;                  - DISK BASE IS POINTED BY DISK POINTER LOCATED
3186                    <1> ;                   AT ABSOLUTE ADDRESS 0:78.
3187                    <1> ;                  - WHEN FORMAT OPERATIONS ARE COMPLETE, THE PARAMETERS
3188                    <1> ;                   SHOULD BE RESTORED TO THEIR RESPECTIVE INITIAL VALUES.
3189                    <1> ;---------------------------------------------------------------------------
3190                    <1> ;     (AH) = 08H READ DRIVE PARAMETERS
3191                    <1> ;     REGISTERS
3192                    <1> ;       INPUT
3193                    <1> ;         (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3194                    <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
3195                    <1> ;            ** EBX = Buffer address for floppy disk parameters table **
3196                    <1> ;       OUTPUT
3197                    <1> ;         (ES:DI) POINTS TO DRIVE PARAMETER TABLE
3198                    <1> ;         *** TRDOS 386 note: floppy disk parameter table (16 bytes)
3199                    <1> ;         will be returned to user in EBX, buffer address *** 27/05/2016 ***
3200                    <1> ;
3201                    <1> ;         (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM NUMBER OF TRACKS
3202                    <1> ;         (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
3203                    <1> ;                BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
3204                    <1> ;         (DH) - MAXIMUM HEAD NUMBER
3205                    <1> ;         (DL) - NUMBER OF DISKETTE DRIVES INSTALLED
3206                    <1> ;         (BH) - 0
3207                    <1> ;         (BL) - BITS 7 THRU 4 - 0
3208                    <1> ;                BITS 3 THRU 0 - VALID DRIVE TYPE VALUE IN CMOS
3209                    <1> ;         (AX) - 0
3210                    <1> ;      UNDER THE FOLLOWING CIRCUMSTANCES:
3211                    <1> ;         (1) THE DRIVE NUMBER IS INVALID,
3212                    <1> ;         (2) THE DRIVE TYPE IS UNKNOWN AND CMOS IS NOT PRESENT,
3213                    <1> ;         (3) THE DRIVE TYPE IS UNKNOWN AND CMOS IS BAD,
3214                    <1> ;         (4) OR THE DRIVE TYPE IS UNKNOWN AND THE CMOS DRIVE TYPE IS INVALID
3215                    <1> ;         THEN ES,AX,BX,CX,DH,DI=0 ; DL=NUMBER OF DRIVES.
3216                    <1> ;         IF NO DRIVES ARE PRESENT THEN: ES,AX,BX,CX,DX,DI=0.
3217                    <1> ;         @DISKETTE_STATUS = 0 AND CY IS RESET.
3218                    <1> ;---------------------------------------------------------------------------
3219                    <1> ;     (AH)= 15H  READ DASD TYPE
3220                    <1> ;     OUTPUT REGISTERS
3221                    <1> ;     (AH) - ON RETURN IF CARRY FLAG NOT SET, OTHERWISE ERROR
3222                    <1> ;            00 - DRIVE NOT PRESENT
3223                    <1> ;            01 - DISKETTE, NO CHANGE LINE AVAILABLE
3224                    <1> ;            02 - DISKETTE, CHANGE LINE AVAILABLE
3225                    <1> ;            03 - RESERVED (FIXED DISK)
3226                    <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3227                    <1> ;---------------------------------------------------------------------------
3228                    <1> ;     (AH)= 16H  DISK CHANGE LINE STATUS
3229                    <1> ;     OUTPUT REGISTERS
3230                    <1> ;     (AH) - 00 - DISK CHANGE LINE NOT ACTIVE
3231                    <1> ;            06 - DISK CHANGE LINE ACTIVE & CARRY BIT ON
3232                    <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3233                    <1> ;---------------------------------------------------------------------------
3234                    <1> ;     (AH)= 17H  SET DASD TYPE FOR FORMAT
3235                    <1> ;     INPUT REGISTERS
3236                    <1> ;     (AL) - 00 - NOT USED
3237                    <1> ;            01 - DISKETTE 320/360K IN 360K DRIVE
3238                    <1> ;            02 - DISKETTE 360K IN 1.2M DRIVE
3239                    <1> ;            03 - DISKETTE 1.2M IN 1.2M DRIVE
3240                    <1> ;            04 - DISKETTE 720K IN 720K DRIVE
3241                    <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED:
3242                    <1> ;             (DO NOT USE WHEN DISKETTE ATTACH CARD USED)
3243                    <1> ;---------------------------------------------------------------------------
3244                    <1> ;     (AH)= 18H  SET MEDIA TYPE FOR FORMAT
3245                    <1> ;     INPUT REGISTERS
3246                    <1> ;     (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM TRACKS
3247                    <1> ;     (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
```

```
                        <1> ;           BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
                        <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHACKED)
                        <1> ;     OUTPUT REGISTERS:
                        <1> ;     (ES:DI) - POINTER TO DRIVE PARAMETERS TABLE FOR THIS MEDIA TYPE,
                        <1> ;              UNCHANGED IF (AH) IS NON-ZERO
                        <1> ;     (AH) - 00H, CY = 0, TRACK AND SECTORS/TRACK COMBINATION IS SUPPORTED
                        <1> ;          - 01H, CY = 1, FUNCTION IS NOT AVAILABLE
                        <1> ;          - 0CH, CY = 1, TRACK AND SECTORS/TRACK COMBINATION IS NOT SUPPORTED
                        <1> ;          - 80H, CY = 1, TIME OUT (DISKETTE NOT PRESENT)
                        <1> ;-------------------------------------------------------------------------------
                        <1> ;     DISK CHANGE STATUS IS ONLY CHECKED WHEN A MEDIA SPECIFIED IS OTHER
                        <1> ;     THAN 360 KB DRIVE. IF THE DISK CHANGE LINE IS FOUND TO BE
                        <1> ;     ACTIVE THE FOLLOWING ACTIONS TAKE PLACE:
                        <1> ;           ATTEMPT TO RESET DISK CHANGE LINE TO INACTIVE STATE.
                        <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> ; 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:
```

```
3353 000042D3 F8                <1>       clc ; 20/07/2020
3354 000042D4 9C                <1>       pushfd
3355 000042D5 0E                <1>       push   cs
3356 000042D6 E801000000        <1>       call   DISKETTE_IO_1
3357 000042DB C3                <1>       retn
3358                            <1>
3359                            <1> DISKETTE_IO_1:
3360                            <1>
3361 000042DC FB                <1>       STI                     ; INTERRUPTS BACK ON
3362 000042DD 55                <1>       PUSH   eBP              ; USER REGISTER
3363 000042DE 57                <1>       PUSH   eDI              ; USER REGISTER
3364 000042DF 52                <1>       PUSH   eDX              ; HEAD #, DRIVE # OR USER REGISTER
3365 000042E0 53                <1>       PUSH   eBX              ; BUFFER OFFSET PARAMETER OR REGISTER
3366 000042E1 51                <1>       PUSH   eCX              ; TRACK #-SECTOR # OR USER REGISTER
3367 000042E2 89E5              <1>       MOV    eBP,eSP                  ; BP     => PARAMETER LIST DEP. ON AH
3368                            <1>                               ; [BP]   = SECTOR #
3369                            <1>                               ; [BP+1] = TRACK #
3370                            <1>                               ; [BP+2] = BUFFER OFFSET
3371                            <1>                               ; FOR RETURN OF DRIVE PARAMETERS:
3372                            <1>                               ; CL/[BP] = BITS 7&6 HI BITS OF MAX CYL
3373                            <1>                               ;          BITS 0-5 MAX SECTORS/TRACK
3374                            <1>                               ; CH/[BP+1] = LOW 8 BITS OF MAX CYL.
3375                            <1>                               ; BL/[BP+2] = BITS 7-4 = 0
3376                            <1>                               ;            BITS 3-0 = VALID CMOS TYPE
3377                            <1>                               ; BH/[BP+3] = 0
3378                            <1>                               ; DL/[BP+4] = # DRIVES INSTALLED
3379                            <1>                               ; DH/[BP+5] = MAX HEAD #
3380                            <1>                               ; DI/[BP+6] = OFFSET TO DISK BASE
3381 000042E4 06                <1>       push   es ; 06/02/2015
3382 000042E5 1E                <1>       PUSH   DS               ; BUFFER SEGMENT PARM OR USER REGISTER
3383 000042E6 56                <1>       PUSH   eSI              ; USER REGISTERS
3384                            <1>       ;CALL  DDS              ; SEGMENT OF BIOS DATA AREA TO DS
3385                            <1>       ;mov   cx, cs
3386                            <1>       ;mov   ds, cx
3387 000042E7 66B91000          <1>       mov    cx, KDATA
3388 000042EB 8ED9              <1>         mov     ds, cx
3389 000042ED 8EC1              <1>         mov     es, cx
3390                            <1>
3391                            <1>       ;CMP   AH,(FNC_TAE-FNC_TAB)/2    ; CHECK FOR > LARGEST FUNCTION
3392 000042EF 80FC19            <1>       cmp    ah,(FNC_TAE-FNC_TAB)/4  ; 18/02/2015
3393 000042F2 7202              <1>       JB     short OK_FUNC       ; FUNCTION OK
3394 000042F4 B414              <1>       MOV    AH,14H              ; REPLACE WITH KNOWN INVALID FUNCTION
3395                            <1> OK_FUNC:
3396 000042F6 80FC01            <1>       CMP    AH,1                ; RESET OR STATUS ?
3397 000042F9 760C              <1>       JBE    short OK_DRV        ; IF RESET OR STATUS DRIVE ALWAYS OK
3398 000042FB 80FC08            <1>       CMP    AH,8                ; READ DRIVE PARMS ?
3399 000042FE 7407              <1>       JZ     short OK_DRV        ; IF SO DRIVE CHECKED LATER
3400 00004300 80FA01            <1>       CMP    DL,1                ; DRIVES 0 AND 1 OK
3401 00004303 7602              <1>       JBE    short OK_DRV        ; IF 0 OR 1 THEN JUMP
3402 00004305 B414              <1>       MOV    AH,14H              ; REPLACE WITH KNOWN INVALID FUNCTION
3403                            <1> OK_DRV:
3404 00004307 31C9              <1>       xor    ecx, ecx
3405                            <1>       ;mov   esi, ecx ; 08/02/2015
3406 00004309 89CF              <1>       mov    edi, ecx ; 08/02/2015
3407 0000430B 88E1              <1>       MOV    CL,AH               ; CL = FUNCTION
3408                            <1>       ;XOR   CH,CH               ; CX = FUNCTION
3409                            <1>       ;SHL   CL, 1               ; FUNCTION TIMES 2
3410 0000430D C0E102            <1>       SHL    CL, 2 ; 20/02/2015  ; FUNCTION TIMES 4 (for 32 bit offset)
3411 00004310 BB[48430000]      <1>       MOV    eBX,FNC_TAB         ; LOAD START OF FUNCTION TABLE
3412 00004315 01CB              <1>       ADD    eBX,eCX                  ; ADD OFFSET INTO TABLE => ROUTINE
3413 00004317 88F4              <1>       MOV    AH,DH               ; AX = HEAD #,# OF SECTORS OR DASD TYPE
3414 00004319 30F6              <1>       XOR    DH,DH               ; DX = DRIVE #
3415 0000431B 6689C6            <1>       MOV    SI,AX               ; SI = HEAD #,# OF SECTORS OR DASD TYPE
3416 0000431E 6689D7            <1>       MOV     DI,DX                   ; DI = DRIVE #
3417                            <1>       ;
3418                            <1>       ; 11/12/2014
3419 00004321 8815[BD6C0000]    <1>         mov     [cfd], dl               ; current floppy drive (for 'GET_PARM')
3420                            <1>       ;
3421 00004327 8A25[78890100]    <1>       MOV    AH, [DSKETTE_STATUS]      ; LOAD STATUS TO AH FOR STATUS FUNCTION
3422 0000432D C605[78890100]00  <1>       MOV    byte [DSKETTE_STATUS],0   ; INITIALIZE FOR ALL OTHERS
3423                            <1>
3424                            <1> ;     THROUGHOUT THE DISKETTE BIOS, THE FOLLOWING INFORMATION IS CONTAINED IN
3425                            <1> ;     THE FOLLOWING MEMORY LOCATIONS AND REGISTERS. NOT ALL DISKETTE BIOS
3426                            <1> ;     FUNCTIONS REQUIRE ALL OF THESE PARAMETERS.
3427                            <1> ;
3428                            <1> ;           DI     : DRIVE #
3429                            <1> ;           SI-HI  : HEAD #
3430                            <1> ;           SI-LOW : # OF SECTORS OR DASD TYPE FOR FORMAT
3431                            <1> ;           ES     : BUFFER SEGMENT
3432                            <1> ;           [BP]   : SECTOR #
3433                            <1> ;           [BP+1] : TRACK #
3434                            <1> ;           [BP+2] : BUFFER OFFSET
3435                            <1> ;
3436                            <1> ;     ACROSS CALLS TO SUBROUTINES THE CARRY FLAG (CY=1), WHERE INDICATED IN
3437                            <1> ;     SUBROUTINE PROLOGUES, REPRESENTS AN EXCEPTION RETURN (NORMALLY AN ERROR
3438                            <1> ;     CONDITION). IN MOST CASES, WHEN CY = 1, @DSKETTE_STATUS CONTAINS THE
3439                            <1> ;     SPECIFIC ERROR CODE.
3440                            <1> ;
3441                            <1>                               ; (AH) = @DSKETTE_STATUS
3442 00004334 FF13              <1>       CALL   dWORD [eBX]        ; CALL THE REQUESTED FUNCTION
3443 00004336 5E                <1>       POP    eSI                ; RESTORE ALL REGISTERS
3444 00004337 1F                <1>       POP    DS
3445 00004338 07                <1>       pop    es     ; 06/02/2015
3446 00004339 59                <1>       POP    eCX
3447 0000433A 5B                <1>       POP    eBX
3448 0000433B 5A                <1>       POP    eDX
3449 0000433C 5F                <1>       POP    eDI
3450 0000433D 89E5              <1>       MOV    eBP, eSP
3451 0000433F 50                <1>       PUSH   eAX
3452 00004340 9C                <1>       PUSHFd
3453 00004341 58                <1>       POP    eAX
3454                            <1>       ;MOV   [BP+6], AX
3455 00004342 89450C            <1>       mov    [ebp+12], eax  ; 18/02/2015, flags
3456 00004345 58                <1>       POP    eAX
3457 00004346 5D                <1>       POP    eBP
```

```
3458 00004347 CF                <1>       IRETd
3459                            <1>
3460                            <1> ;--------------------------------------------------------------------------------
3461                            <1> ; DW --> dd (06/02/2015)
3462 00004348 [AC430000]        <1> FNC_TAB      dd     DSK_RESET           ; AH = 00H; RESET
3463 0000434C [22440000]        <1>       dd     DSK_STATUS          ; AH = 01H; STATUS
3464 00004350 [33440000]        <1>       dd     DSK_READ            ; AH = 02H; READ
3465 00004354 [44440000]        <1>       dd     DSK_WRITE           ; AH = 03H; WRITE
3466 00004358 [55440000]        <1>       dd     DSK_VERF            ; AH = 04H; VERIFY
3467 0000435C [66440000]        <1>       dd     DSK_FORMAT          ; AH = 05H; FORMAT
3468 00004360 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 06H; INVALID
3469 00004364 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 07H; INVALID
3470 00004368 [F8440000]        <1>       dd     DSK_PARMS           ; AH = 08H; READ DRIVE PARAMETERS
3471 0000436C [EB440000]        <1>       dd     FNC_ERR                   ; AH = 09H; INVALID
3472 00004370 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 0AH; INVALID
3473 00004374 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 0BH; INVALID
3474 00004378 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 0CH; INVALID
3475 0000437C [EB440000]        <1>       dd     FNC_ERR                   ; AH = 0DH; INVALID
3476 00004380 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 0EH; INVALID
3477 00004384 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 0FH; INVALID
3478 00004388 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 10H; INVALID
3479 0000438C [EB440000]        <1>       dd     FNC_ERR                   ; AH = 11H; INVALID
3480 00004390 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 12H; INVALID
3481 00004394 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 13H; INVALID
3482 00004398 [EB440000]        <1>       dd     FNC_ERR                   ; AH = 14H; INVALID
3483 0000439C [DE450000]        <1>       dd     DSK_TYPE            ; AH = 15H; READ DASD TYPE
3484 000043A0 [0C460000]        <1>       dd     DSK_CHANGE          ; AH = 16H; CHANGE STATUS
3485 000043A4 [46460000]        <1>       dd     FORMAT_SET          ; AH = 17H; SET DASD TYPE
3486 000043A8 [C0460000]        <1>       dd     SET_MEDIA           ; AH = 18H; SET MEDIA TYPE
3487                            <1> FNC_TAE EQU     $                       ; END
3488                            <1>
3489                            <1> ;--------------------------------------------------------------------------------
3490                            <1> ; DISK RESET (AH = 00H)
3491                            <1> ;           RESET THE DISKETTE SYSTEM.
3492                            <1> ;
3493                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3494                            <1> ;--------------------------------------------------------------------------------
3495                            <1> DSK_RESET:
3496 000043AC 66BAF203          <1>       MOV    DX,03F2H            ; ADAPTER CONTROL PORT
3497 000043B0 FA                <1>       CLI                        ; NO INTERRUPTS
3498 000043B1 A0[76890100]      <1>       MOV    AL,[MOTOR_STATUS]   ; GET DIGITAL OUTPUT REGISTER REFLECTION
3499 000043B6 243F              <1>       AND    AL,00111111B        ; KEEP SELECTED AND MOTOR ON BITS
3500 000043B8 C0C004            <1>       ROL    AL,4                ; MOTOR VALUE TO HIGH NIBBLE
3501                            <1>                                  ; DRIVE SELECT TO LOW NIBBLE
3502 000043BB 0C08              <1>       OR     AL,00001000B        ; TURN ON INTERRUPT ENABLE
3503 000043BD EE                <1>       OUT    DX,AL               ; RESET THE ADAPTER
3504 000043BE C605[75890100]00  <1>       MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
3505                            <1>       ;JMP   $+2                 ; WAIT FOR I/O
3506                            <1>       ;JMP   $+2                 ; WAIT FOR I/O (TO INSURE MINIMUM
3507                            <1>                                  ;      PULSE WIDTH)
3508                            <1>       ; 19/12/2014
3509                            <1>       NEWIODELAY
2973 000043C5 E6EB              <2>  out 0ebh,al
3510                            <1>
3511                            <1>       ; 17/12/2014
3512                            <1>       ; AWARD BIOS 1999 - RESETDRIVES (ADISK.ASM)
3513 000043C7 B915000000        <1>       mov    ecx, WAITCPU_RESET_ON     ; cx = 21 -- Min. 14 micro seconds !?
3514                            <1> wdw1:
3515                            <1>       NEWIODELAY   ; 27/02/2015
2973 000043CC E6EB              <2>  out 0ebh,al
3516 000043CE E2FC              <1>       loop   wdw1
3517                            <1>       ;
3518 000043D0 0C04              <1>       OR     AL,00000100B        ; TURN OFF RESET BIT
3519 000043D2 EE                <1>       OUT    DX,AL               ; RESET THE ADAPTER
3520                            <1>       ; 16/12/2014
3521                            <1>       IODELAY
2968 000043D3 EB00              <2>  jmp short $+2
2969 000043D5 EB00              <2>  jmp short $+2
3522                            <1>       ;
3523                            <1>       ;STI                       ; ENABLE THE INTERRUPTS
3524 000043D7 E8210C0000        <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
3525 000043DC 723B              <1>       JC     short DR_ERR        ; IF ERROR, RETURN IT
3526 000043DE 66B9C000          <1>       MOV    CX,11000000B        ; CL = EXPECTED @NEC_STATUS
3527                            <1> NXT_DRV:
3528                            <1>       ;PUSH  CX                  ; SAVE FOR CALL
3529                            <1>       ; 11/04/2021
3530 000043E2 51                <1>       push   ecx
3531 000043E3 B8[18440000]      <1>       MOV    eAX, DR_POP_ERR     ; LOAD NEC_OUTPUT ERROR ADDRESS
3532 000043E8 50                <1>       PUSH   eAX                 ; "
3533 000043E9 B408              <1>       MOV    AH,08H              ; SENSE INTERRUPT STATUS COMMAND
3534 000043EB E8020B0000        <1>       CALL   NEC_OUTPUT
3535 000043F0 58                <1>       POP    eAX                 ; THROW AWAY ERROR RETURN
3536 000043F1 E8370C0000        <1>       CALL   RESULTS                   ; READ IN THE RESULTS
3537                            <1>       ;POP   CX                  ; RESTORE AFTER CALL
3538                            <1>       ; 11/04/2021
3539 000043F6 59                <1>       pop    ecx
3540 000043F7 7220              <1>       JC     short DR_ERR        ; ERROR RETURN
3541 000043F9 3A0D[79890100]    <1>       CMP    CL, [NEC_STATUS]    ; TEST FOR DRIVE READY TRANSITION
3542 000043FF 7518              <1>       JNZ    short DR_ERR        ; EVERYTHING OK
3543 00004401 FEC1              <1>       INC    CL                  ; NEXT EXPECTED @NEC_STATUS
3544 00004403 80F9C3            <1>       CMP    CL,11000011B        ; ALL POSSIBLE DRIVES CLEARED
3545 00004406 76DA              <1>       JBE    short NXT_DRV       ; FALL THRU IF 11000100B OR >
3546                            <1>       ;
3547 00004408 E86E030000        <1>       CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
3548                            <1> RESBAC:
3549 0000440D E80C090000        <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
3550 00004412 6689F3            <1>       MOV    BX,SI               ; GET SAVED AL TO BL
3551 00004415 88D8              <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
3552 00004417 C3                <1>       RETn
3553                            <1> DR_POP_ERR:
3554                            <1>       ;POP   CX                  ; CLEAR STACK
3555                            <1>       ; 11/04/2021
3556 00004418 59                <1>       pop    ecx
3557                            <1> DR_ERR:
3558 00004419 800D[78890100]20  <1>       OR     byte [DSKETTE_STATUS],BAD_NEC ; SET ERROR CODE
```

```
3559 00004420 EBEB                <1>       JMP    SHORT RESBAC       ; RETURN FROM RESET
3560                              <1>
3561                              <1> ;--------------------------------------------------------------------------------
3562                              <1> ; DISK_STATUS      (AH = 01H)
3563                              <1> ;     DISKETTE STATUS.
3564                              <1> ;
3565                              <1> ; ON ENTRY:  AH : STATUS OF PREVIOUS OPERATION
3566                              <1> ;
3567                              <1> ; ON EXIT:   AH, @DSKETTE_STATUS, CY REFLECT STATUS OF PREVIOUS OPERATION.
3568                              <1> ;--------------------------------------------------------------------------------
3569                              <1> DSK_STATUS:
3570 00004422 8825[78890100]      <1>       MOV    [DSKETTE_STATUS],AH ; PUT BACK FOR SETUP END
3571 00004428 E8F1080000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
3572 0000442D 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
3573 00004430 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
3574 00004432 C3                  <1>       RETn
3575                              <1>
3576                              <1> ;--------------------------------------------------------------------------------
3577                              <1> ; DISK_READ  (AH = 02H)
3578                              <1> ;     DISKETTE READ.
3579                              <1> ;
3580                              <1> ; ON ENTRY:  DI     : DRIVE #
3581                              <1> ;            SI-HI  : HEAD #
3582                              <1> ;            SI-LOW : # OF SECTORS
3583                              <1> ;            ES     : BUFFER SEGMENT
3584                              <1> ;            [BP]   : SECTOR #
3585                              <1> ;            [BP+1] : TRACK #
3586                              <1> ;            [BP+2] : BUFFER OFFSET
3587                              <1> ;
3588                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3589                              <1> ;--------------------------------------------------------------------------------
3590                              <1>
3591                              <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
3592                              <1>
3593                              <1> DSK_READ:
3594 00004433 8025[76890100]7F    <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
3595 0000443A 66B846E6            <1>       MOV    AX,0E646H           ; AX = NEC COMMAND, DMA COMMAND
3596 0000443E E842040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
3597 00004443 C3                  <1>       RETn
3598                              <1>
3599                              <1> ;--------------------------------------------------------------------------------
3600                              <1> ; DISK_WRITE (AH = 03H)
3601                              <1> ;     DISKETTE WRITE.
3602                              <1> ;
3603                              <1> ; ON ENTRY:  DI     : DRIVE #
3604                              <1> ;            SI-HI  : HEAD #
3605                              <1> ;            SI-LOW : # OF SECTORS
3606                              <1> ;            ES     : BUFFER SEGMENT
3607                              <1> ;            [BP]   : SECTOR #
3608                              <1> ;            [BP+1] : TRACK #
3609                              <1> ;            [BP+2] : BUFFER OFFSET
3610                              <1> ;
3611                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3612                              <1> ;--------------------------------------------------------------------------------
3613                              <1>
3614                              <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
3615                              <1>
3616                              <1> DSK_WRITE:
3617 00004444 66B84AC5            <1>       MOV    AX,0C54AH           ; AX = NEC COMMAND, DMA COMMAND
3618 00004448 800D[76890100]80    <1>         OR     byte [MOTOR_STATUS],10000000B ; INDICATE WRITE OPERATION
3619 0000444F E831040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
3620 00004454 C3                  <1>       RETn
3621                              <1>
3622                              <1> ;--------------------------------------------------------------------------------
3623                              <1> ; DISK_VERF  (AH = 04H)
3624                              <1> ;     DISKETTE VERIFY.
3625                              <1> ;
3626                              <1> ; ON ENTRY:  DI     : DRIVE #
3627                              <1> ;            SI-HI  : HEAD #
3628                              <1> ;            SI-LOW : # OF SECTORS
3629                              <1> ;            ES     : BUFFER SEGMENT
3630                              <1> ;            [BP]   : SECTOR #
3631                              <1> ;            [BP+1] : TRACK #
3632                              <1> ;            [BP+2] : BUFFER OFFSET
3633                              <1> ;
3634                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3635                              <1> ;--------------------------------------------------------------------------------
3636                              <1> DSK_VERF:
3637 00004455 8025[76890100]7F    <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
3638 0000445C 66B842E6            <1>       MOV    AX,0E642H           ; AX = NEC COMMAND, DMA COMMAND
3639 00004460 E820040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
3640 00004465 C3                  <1>       RETn
3641                              <1>
3642                              <1> ;--------------------------------------------------------------------------------
3643                              <1> ; DISK_FORMAT      (AH = 05H)
3644                              <1> ;     DISKETTE FORMAT.
3645                              <1> ;
3646                              <1> ; ON ENTRY:  DI     : DRIVE #
3647                              <1> ;            SI-HI  : HEAD #
3648                              <1> ;            SI-LOW : # OF SECTORS
3649                              <1> ;            ES     : BUFFER SEGMENT
3650                              <1> ;            [BP]   : SECTOR #
3651                              <1> ;            [BP+1] : TRACK #
3652                              <1> ;            [BP+2] : BUFFER OFFSET
3653                              <1> ;            @DISK_POINTER POINTS TO THE PARAMETER TABLE OF THIS DRIVE
3654                              <1> ;
3655                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3656                              <1> ;--------------------------------------------------------------------------------
3657                              <1> DSK_FORMAT:
3658 00004466 E859030000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
3659 0000446B E849050000          <1>       CALL   FMT_INIT            ; ESTABLISH STATE IF UNESTABLISHED
3660 00004470 800D[76890100]80    <1>         OR     byte [MOTOR_STATUS], 10000000B ; INDICATE WRITE OPERATION
3661 00004477 E891050000          <1>       CALL   MED_CHANGE          ; CHECK MEDIA CHANGE AND RESET IF SO
3662 0000447C 725D                <1>         JC     short FM_DON            ; MEDIA CHANGED, SKIP
3663 0000447E E8F8020000          <1>       CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
```

```
3664 00004483 E8F5050000          <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMPT RATE IS SAME AS LAST RATE
3665 00004488 7405                <1>         JZ     short FM_WR             ; YES, SKIP SPECIFY COMMAND
3666 0000448A E8CE050000          <1>       CALL   SEND_RATE           ; SEND DATA RATE TO CONTROLLER
3667                              <1> FM_WR:
3668 0000448F E87E060000          <1>       CALL   FMTDMA_SET          ; SET UP THE DMA FOR FORMAT
3669 00004494 7245                <1>         JC     short FM_DON            ; RETURN WITH ERROR
3670 00004496 B44D                <1>       MOV    AH,04DH                ; ESTABLISH THE FORMAT COMMAND
3671 00004498 E8D9060000          <1>       CALL   NEC_INIT            ; INITIALIZE THE NEC
3672 0000449D 723C                <1>         JC     short FM_DON            ; ERROR - EXIT
3673 0000449F B8[DB440000]        <1>         MOV    eAX, FM_DON             ; LOAD ERROR ADDRESS
3674 000044A4 50                  <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
3675 000044A5 B203                <1>       MOV    DL,3                ; BYTES/SECTOR VALUE TO NEC
3676 000044A7 E840090000          <1>       CALL   GET_PARM
3677 000044AC E8410A0000          <1>       CALL   NEC_OUTPUT
3678 000044B1 B204                <1>       MOV    DL,4                ; SECTORS/TRACK VALUE TO NEC
3679 000044B3 E834090000          <1>       CALL   GET_PARM
3680 000044B8 E8350A0000          <1>       CALL   NEC_OUTPUT
3681 000044BD B207                <1>       MOV    DL,7                ; GAP LENGTH VALUE TO NEC
3682 000044BF E828090000          <1>       CALL   GET_PARM
3683 000044C4 E8290A0000          <1>       CALL   NEC_OUTPUT
3684 000044C9 B208                <1>       MOV    DL,8                ; FILLER BYTE TO NEC
3685 000044CB E81C090000          <1>       CALL   GET_PARM
3686 000044D0 E81D0A0000          <1>       CALL   NEC_OUTPUT
3687 000044D5 58                  <1>       POP    eAX                 ; THROW AWAY ERROR
3688 000044D6 E817070000          <1>       CALL   NEC_TERM            ; TERMINATE, RECEIVE STATUS, ETC,
3689                              <1> FM_DON:
3690 000044DB E815030000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
3691 000044E0 E839080000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
3692 000044E5 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
3693 000044E8 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
3694 000044EA C3                  <1>       RETn
3695                              <1>
3696                              <1> ;--------------------------------------------------------------------------
3697                              <1> ; FNC_ERR
3698                              <1> ;     INVALID FUNCTION REQUESTED OR INVALID DRIVE:
3699                              <1> ;     SET BAD COMMAND IN STATUS.
3700                              <1> ;
3701                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3702                              <1> ;--------------------------------------------------------------------------
3703                              <1> FNC_ERR:                       ; INVALID FUNCTION REQUEST
3704 000044EB 6689F0              <1>       MOV    AX,SI               ; RESTORE AL
3705 000044EE B401                <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
3706 000044F0 8825[78890100]      <1>       MOV    [DSKETTE_STATUS],AH ; STORE IN DATA AREA
3707 000044F6 F9                  <1>       STC                        ; SET CARRY INDICATING ERROR
3708 000044F7 C3                  <1>       RETn
3709                              <1>
3710                              <1> ; 30/08/2020
3711                              <1> ; 29/08/2020
3712                              <1> ; 01/06/2016
3713                              <1> ; 28/05/2016
3714                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v.2.0)
3715                              <1> ;--------------------------------------------------------------------------
3716                              <1> ; DISK_PARMS (AH = 08H)
3717                              <1> ;     READ DRIVE PARAMETERS.
3718                              <1> ;
3719                              <1> ; ON ENTRY:  DI : DRIVE #
3720                              <1> ;            ; 27/05/2016
3721                              <1> ;            EBX = Buffer Address for floppy disk parameters table (16 bytes)
3722                              <1> ;
3723                              <1> ; ON EXIT:   CL/[BP]   = BITS 7 & 6 HI 2 BITS OF MAX CYLINDER
3724                              <1> ;                        BITS 0-5 MAX SECTORS/TRACK
3725                              <1> ;            CH/[BP+1] = LOW 8 BITS OF MAX CYLINDER
3726                              <1> ;            BL/[BP+2] = BITS 7-4 = 0
3727                              <1> ;                        BITS 3-0 = VALID CMOS DRIVE TYPE
3728                              <1> ;            BH/[BP+3] = 0
3729                              <1> ;            DL/[BP+4] = # DRIVES INSTALLED (VALUE CHECKED)
3730                              <1> ;            DH/[BP+5] = MAX HEAD #
3731                              <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
3732                              <1> ;            ** EBX = Buffer address for floppy disk parameters table **
3733                              <1> ;            ;DI/[BP+6] = OFFSET TO DISK_BASE
3734                              <1> ;            ;ES        = SEGMENT OF DISK_BASE
3735                              <1> ;
3736                              <1> ;            AX        = 0
3737                              <1> ;
3738                              <1> ;            NOTE : THE ABOVE INFORMATION IS STORED IN THE USERS STACK AT
3739                              <1> ;                   THE LOCATIONS WHERE THE MAIN ROUTINE WILL POP THEM
3740                              <1> ;                   INTO THE APPROPRIATE REGISTERS BEFORE RETURNING TO THE
3741                              <1> ;                   CALLER.
3742                              <1> ;--------------------------------------------------------------------------
3743                              <1> DSK_PARMS:
3744 000044F8 E8C7020000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH,
3745                              <1>      ;       MOV    WORD [BP+2],0       ; DRIVE TYPE = 0
3746                              <1>      ;  MOV     AX, [EQUIP_FLAG]        ; LOAD EQUIPMENT FLAG FOR # DISKETTES
3747                              <1>      ;  AND     AL,11000001B            ; KEEP DISKETTE DRIVE BITS
3748                              <1>      ;  MOV     DL,2                    ; DISKETTE DRIVES = 2
3749                              <1>      ;  CMP     AL,01000001B            ; 2 DRIVES INSTALLED ?
3750                              <1>      ;  JZ      short STO_DL            ; IF YES JUMP
3751                              <1>      ;  DEC     DL                      ; DISKETTE DRIVES = 1
3752                              <1>      ;  CMP     AL,00000001B            ; 1 DRIVE INSTALLED ?
3753                              <1>      ;  JNZ     short NON_DRV           ; IF NO JUMP
3754 000044FD 29D2                <1>       sub    edx, edx
3755 000044FF 66A1[CE6C0000]      <1>       mov     ax, [fd0_type]
3756 00004505 6621C0              <1>       and     ax, ax
3757 00004508 0F8491000000        <1>         jz      NON_DRV
3758 0000450E FEC2                <1>       inc     dl
3759 00004510 20E4                <1>       and     ah, ah
3760 00004512 7402                <1>       jz      short STO_DL
3761 00004514 FEC2                <1>       inc     dl
3762                              <1> STO_DL:
3763                              <1>       ; 30/08/2020
3764 00004516 6639FA              <1>       cmp    dx, di
3765 00004519 0F8680000000        <1>       jna    NON_DRV
3766                              <1>       ;
3767                              <1>       ;MOV   [BP+4],DL           ; STORE NUMBER OF DRIVES
3768 0000451F 895508              <1>       mov    [ebp+8], edx ; 20/02/2015
```

```
3769                            <1>       ; 11/04/2021
3770                            <1>       ;CMP   DI,1               ; CHECK FOR VALID DRIVE
3771                            <1>       ;;JA   short NON_DRV1           ; DRIVE INVALID
3772                            <1>       ;ja    NON_DRV1 ; 29/08/2020
3773                            <1>       ;
3774                            <1>       ;MOV   BYTE [BP+5],1      ; MAXIMUM HEAD NUMBER =   1
3775 00004522 C6450901         <1>       mov    byte [ebp+9], 1  ; 20/02/2015
3776 00004526 E8B8080000       <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN AL
3777                            <1>       ;;20/02/2015
3778                            <1>       ;;JC   short CHK_EST      ; IF CMOS BAD CHECKSUM ESTABLISHED
3779                            <1>       ;;OR   AL,AL              ; TEST FOR NO DRIVE TYPE
3780 0000452B 740F             <1>       JZ     short CHK_EST      ; JUMP IF SO
3781 0000452D E820020000       <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
3782 00004532 7208             <1>       JC     short CHK_EST      ; TYPE NOT IN TABLE (POSSIBLE BAD CMOS)
3783                            <1>       ;MOV   [BP+2],AL          ; STORE VALID CMOS DRIVE TYPE
3784                            <1>         ;mov [ebp+4], al ; 06/02/2015
3785 00004534 8A4B04           <1>       MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
3786 00004537 8A6B0B           <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
3787 0000453A EB36             <1>       JMP    SHORT STO_CX       ; CMOS GOOD, USE CMOS
3788                            <1> CHK_EST:
3789 0000453C 8AA7[85890100]   <1>       MOV    AH, [DSK_STATE+eDI] ; LOAD STATE FOR THIS DRIVE
3790 00004542 F6C410           <1>       TEST   AH,MED_DET         ; CHECK FOR ESTABLISHED STATE
3791 00004545 745B             <1>       JZ     short NON_DRV1           ; CMOS BAD/INVALID OR UNESTABLISHED
3792                            <1> USE_EST:
3793 00004547 80E4C0           <1>       AND    AH,RATE_MSK        ; ISOLATE STATE
3794 0000454A 80FC80           <1>       CMP    AH,RATE_250        ; RATE 250 ?
3795 0000454D 757B             <1>       JNE    short USE_EST2           ; NO, GO CHECK OTHER RATE
3796                            <1>
3797                            <1> ;-----       DATA RATE IS 250 KBS, TRY 360 KB TABLE FIRST
3798                            <1>
3799 0000454F B001             <1>       MOV    AL,01              ; DRIVE TYPE 1 (360KB)
3800 00004551 E8FC010000       <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
3801 00004556 8A4B04           <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
3802 00004559 8A6B0B           <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
3803 0000455C F687[85890100]01 <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; 80 TRACK ?
3804 00004563 740D             <1>       JZ     short STO_CX       ; MUST BE 360KB DRIVE
3805                            <1>
3806                            <1> ;-----       IT IS 1.44 MB DRIVE
3807                            <1>
3808                            <1> PARM144:
3809 00004565 B004             <1>       MOV    AL,04              ; DRIVE TYPE 4 (1.44MB)
3810 00004567 E8E6010000       <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
3811 0000456C 8A4B04           <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
3812 0000456F 8A6B0B           <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
3813                            <1> STO_CX:
3814 00004572 894D00           <1>       MOV    [eBP],eCX          ; SAVE POINTER IN STACK FOR RETURN
3815                            <1> ES_DI:
3816                            <1>       ;MOV   [BP+6],BX          ; ADDRESS OF MEDIA/DRIVE PARM TABLE
3817                            <1>       ;mov   [ebp+12], ebx ; 06/02/2015
3818                            <1>       ;MOV   AX,CS              ; SEGMENT MEDIA/DRIVE PARAMETER TABLE
3819                            <1>       ;MOV   ES,AX              ; ES IS SEGMENT OF TABLE
3820                            <1>       ;
3821                            <1>       ; 28/05/2016
3822                            <1>       ; 27/05/2016
3823                            <1>       ; return floppy disk parameters table to user
3824                            <1>       ; in user's buffer, which is pointed by EBX
3825                            <1>       ;
3826 00004575 57               <1>       push   edi
3827 00004576 8B7D04           <1>       mov    edi, [ebp+4]             ; ebx (input), user's buffer address
3828                            <1>       ; 29/08/2020
3829 00004579 09FF             <1>       or     edi, edi
3830 0000457B 7417             <1>       jz     short no_copy_fdpt
3831                            <1>       ;
3832 0000457D 0FB6C0           <1>       movzx  eax, al
3833 00004580 894504           <1>         mov  [ebp+4], eax   ; ebx      ; drive type (for floppy drives)
3834                            <1>       ; 01/06/2016 (INT 33h, disk type return for floppy disks, in BL)
3835 00004583 A3[7C950100]     <1>       mov    [user_buffer], eax  ; 01/06/2016 (overwrite ebx return value)
3836                            <1>       ;(INT 33h, Function 08h will replace user's buffer addr with disk type!)
3837                            <1>       ;
3838 00004588 89DE             <1>       mov    esi, ebx           ; floppy disk parameter table (16 bytes)
3839 0000458A B910000000       <1>       mov    ecx, 16 ; 16 bytes
3840 0000458F E890D40000       <1>         call    transfer to user buffer ; trdosk6.s (16/05/2016)
3841                            <1> no_copy_fdpt:
3842 00004594 5F               <1>       pop    edi
3843                            <1> DP_OUT:
3844 00004595 E85B020000       <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
3845 0000459A 6631C0           <1>       XOR    AX,AX              ; CLEAR
3846 0000459D F8               <1>       CLC
3847 0000459E C3               <1>       RETn
3848                            <1>
3849                            <1> ;-----       NO DRIVE PRESENT HANDLER
3850                            <1>
3851                            <1> NON_DRV:
3852                            <1>       ;MOV   BYTE [BP+4],0      ; CLEAR NUMBER OF DRIVES
3853 0000459F 895508           <1>       mov    [ebp+8], edx ; 0 ; 20/02/2015
3854                            <1> NON_DRV1:
3855 000045A2 6681FF8000       <1>       CMP    DI,80H             ; CHECK FOR FIXED MEDIA TYPE REQUEST
3856 000045A7 720C             <1>       JB     short NON_DRV2           ; CONTINUE IF NOT REQUEST FALL THROUGH
3857                            <1>
3858                            <1> ;-----       FIXED DISK REQUEST FALL THROUGH ERROR
3859                            <1>
3860 000045A9 E847020000       <1>       CALL   XLAT_OLD           ; ELSE TRANSLATE TO COMPATIBLE MODE
3861 000045AE 6689F0           <1>       MOV    AX,SI              ; RESTORE AL
3862 000045B1 B401             <1>       MOV    AH,BAD_CMD         ; SET BAD COMMAND ERROR
3863 000045B3 F9               <1>       STC
3864 000045B4 C3               <1>       RETn
3865                            <1>
3866                            <1> NON_DRV2:
3867                            <1>       ;XOR   AX,AX              ; CLEAR PARMS IF NO DRIVES OR CMOS BAD
3868 000045B5 31C0             <1>       xor    eax, eax
3869 000045B7 66894500         <1>       MOV    [eBP],AX           ; TRACKS, SECTORS/TRACK = 0
3870                            <1>       ;MOV   [BP+5],AH          ; HEAD = 0
3871 000045BB 886509           <1>       mov    [ebp+9], ah ; 06/02/2015
3872                            <1>       ;MOV   [BP+6],AX          ; OFFSET TO DISK_BASE = 0
3873 000045BE 89450C           <1>       mov    [ebp+12], eax
```

```
                        <1>       ;;MOV  ES,AX              ; ES IS SEGMENT OF TABLE
                        <1>       ;JMP   SHORT DP_OUT
                        <1>
                        <1>       ; 30/08/2020
000045C1 E82F020000     <1>       call   XLAT_OLD
                        <1>       ;mov   ah, NOT_RDY ; drive not ready
000045C6 B407           <1>       mov    ah, INIT_FAIL ; DRIVE PARAMETER ACTIVITY FAILED
000045C8 F9             <1>       stc    ; cf -> 1, ah = 'drive not ready' error code
000045C9 C3             <1>       retn
                        <1>
                        <1> ;-----      DATA RATE IS EITHER 300 KBS OR 500 KBS, TRY 1.2 MB TABLE FIRST
                        <1>
                        <1> USE_EST2:
000045CA B002           <1>       MOV    AL,02              ; DRIVE TYPE 2 (1.2MB)
000045CC E881010000     <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
000045D1 8A4B04         <1>         MOV     CL, [eBX+MD.SEC_TRK]    ; GET SECTOR/TRACK
000045D4 8A6B0B         <1>         MOV     CH, [eBX+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
000045D7 80FC40         <1>       CMP    AH,RATE_300        ; RATE 300 ?
000045DA 7496           <1>       JZ     short STO_CX       ; MUST BE 1.2MB DRIVE
000045DC EB87           <1>       JMP    SHORT PARM144      ; ELSE, IT IS 1.44MB DRIVE
                        <1>
                        <1> ; 30/08/2020
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; DISK_TYPE (AH = 15H)
                        <1> ;     THIS ROUTINE RETURNS THE TYPE OF MEDIA INSTALLED.
                        <1> ;
                        <1> ;  ON ENTRY: DI = DRIVE #
                        <1> ;
                        <1> ;  ON EXIT:  AH = DRIVE TYPE, CY=0
                        <1> ;--------------------------------------------------------------------------
                        <1> DSK_TYPE:
000045DE E8E1010000     <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH.
000045E3 8A87[85890100] <1>       MOV    AL,[DSK_STATE+eDI] ; GET PRESENT STATE INFORMATION
000045E9 08C0           <1>       OR     AL,AL              ; CHECK FOR NO DRIVE
000045EB 7416           <1>       JZ     short NO_DRV
000045ED B401           <1>       MOV    AH,NOCHGLN         ; NO CHANGE LINE FOR 40 TRACK DRIVE
000045EF A801           <1>       TEST   AL,TRK_CAPA        ; IS THIS DRIVE AN 80 TRACK DRIVE?
000045F1 7402           <1>       JZ     short DT_BACK      ; IF NO JUMP
000045F3 B402           <1>       MOV    AH,CHGLN           ; CHANGE LINE FOR 80 TRACK DRIVE
                        <1> DT_BACK:
                        <1>       ;PUSH  AX                 ; SAVE RETURN VALUE
                        <1>       ; 11/04/2021
000045F5 50             <1>       push   eax
000045F6 E8FA010000     <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
                        <1>       ; 11/04/2021
000045FB 58             <1>       pop    eax
000045FC F8             <1>       CLC                       ; NO ERROR
000045FD 6689F3         <1>       MOV    BX,SI              ; GET SAVED AL TO BL
00004600 88D8           <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
00004602 C3             <1>       RETn
                        <1> NO_DRV:
                        <1>       ;XOR   AH,AH              ; NO DRIVE PRESENT OR UNKNOWN
                        <1>       ;JMP   SHORT DT_BACK
                        <1>
                        <1>       ; 30/08/2020
00004603 E8ED010000     <1>       call   XLAT_OLD
00004608 29C0           <1>       sub    eax, eax
0000460A F9             <1>       stc    ; cf = 1 -> drive not ready, ah = 0 (disk type = 0)
0000460B C3             <1>       retn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; DISK_CHANGE      (AH = 16H)
                        <1> ;     THIS ROUTINE RETURNS THE STATE OF THE DISK CHANGE LINE.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE #
                        <1> ;
                        <1> ; ON EXIT:   AH = @DSKETTE_STATUS
                        <1> ;                 00 - DISK CHANGE LINE INACTIVE, CY = 0
                        <1> ;                 06 - DISK CHANGE LINE ACTIVE, CY = 1
                        <1> ;--------------------------------------------------------------------------
                        <1> DSK_CHANGE:
0000460C E8B3010000     <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH.
00004611 8A87[85890100] <1>       MOV    AL, [DSK_STATE+eDI] ; GET MEDIA STATE INFORMATION
00004617 08C0           <1>       OR     AL,AL              ; DRIVE PRESENT ?
00004619 7422           <1>       JZ     short DC_NON       ; JUMP IF NO DRIVE
0000461B A801           <1>       TEST   AL,TRK_CAPA        ; 80 TRACK DRIVE ?
0000461D 7407           <1>       JZ     short SETIT        ; IF SO , CHECK CHANGE LINE
                        <1> DC0:
0000461F E8640A0000     <1>         CALL    READ_DSKCHNG            ; GO CHECK STATE OF DISK CHANGE LINE
00004624 7407           <1>       JZ     short FINIS        ; CHANGE LINE NOT ACTIVE
                        <1>
00004626 C605[78890100]06 <1> SETIT:      MOV    byte [DSKETTE_STATUS], MEDIA_CHANGE ; INDICATE MEDIA REMOVED
                        <1>
0000462D E8C3010000     <1> FINIS:      CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
00004632 E8E7060000     <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
00004637 6689F3         <1>       MOV    BX,SI              ; GET SAVED AL TO BL
0000463A 88D8           <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
0000463C C3             <1>       RETn
                        <1> DC_NON:
0000463D 800D[78890100]80 <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; SET TIMEOUT, NO DRIVE
00004644 EBE7           <1>       JMP    SHORT FINIS
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; FORMAT_SET (AH = 17H)
                        <1> ;     THIS ROUTINE IS USED TO ESTABLISH THE TYPE OF MEDIA TO BE USED
                        <1> ;     FOR THE FOLLOWING FORMAT OPERATION.
                        <1> ;
                        <1> ; ON ENTRY:  SI LOW = DASD TYPE FOR FORMAT
                        <1> ;            DI     = DRIVE #
                        <1> ;
                        <1> ; ON EXIT:   @DSKETTE_STATUS REFLECTS STATUS
                        <1> ;            AH = @DSKETTE_STATUS
                        <1> ;            CY = 1 IF ERROR
                        <1> ;--------------------------------------------------------------------------
```

```
                                <1> FORMAT_SET:
00004646 E879010000             <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH.
                                <1>       ;PUSH  SI                 ; SAVE DASD TYPE
                                <1>       ; 11/04/2021
0000464B 56                     <1>       push   esi
                                <1>       ;MOV   AX,SI              ; AH = ? , AL = DASD TYPE
                                <1>       ;XOR   AH,AH              ; AH = 0 , AL = DASD TYPE
                                <1>       ;MOV   SI,AX              ; SI = DASD TYPE
                                <1>       ; 11/04/2021
0000464C 89F0                   <1>       mov    eax, esi
0000464E 0FB6F0                 <1>       movzx  esi, al
00004651 80A7[85890100]0F       <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
                                <1>       ;DEC   SI                 ; CHECK FOR 320/360K MEDIA & DRIVE
                                <1>       ; 11/04/2021
00004658 4E                     <1>       dec    esi
00004659 7509                   <1>       JNZ    short NOT_320      ; BYPASS IF NOT
0000465B 808F[85890100]90       <1>       OR     byte [DSK_STATE+eDI], MED_DET+RATE_250 ; SET TO 320/360
00004662 EB45                   <1>       JMP    SHORT S0
                                <1>
                                <1> NOT_320:
00004664 E8A4030000             <1>       CALL   MED_CHANGE         ; CHECK FOR TIME_OUT
00004669 803D[78890100]80       <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT
00004670 7437                   <1>       JZ     short S0           ; IF TIME OUT TELL CALLER
                                <1> S3:
                                <1>       ;DEC   SI                 ; CHECK FOR 320/360K IN 1.2M DRIVE
                                <1>       ; 11/04/2021
00004672 4E                     <1>       dec    esi
00004673 7509                   <1>       JNZ    short NOT_320_12   ; BYPASS IF NOT
00004675 808F[85890100]70       <1>       OR     byte [DSK_STATE+eDI], MED_DET+DBL_STEP+RATE_300 ; SET STATE
0000467C EB2B                   <1>       JMP    SHORT S0
                                <1>
                                <1> NOT_320_12:
                                <1>       ;DEC   SI                 ; CHECK FOR 1.2M MEDIA IN 1.2M DRIVE
                                <1>       ; 11/04/2021
0000467E 4E                     <1>       dec    esi
0000467F 7509                   <1>       JNZ    short NOT_12       ; BYPASS IF NOT
00004681 808F[85890100]10       <1>       OR     byte [DSK_STATE+eDI], MED_DET+RATE_500 ; SET STATE VARIABLE
00004688 EB1F                   <1>       JMP    SHORT S0           ; RETURN TO CALLER
                                <1>
                                <1> NOT_12:
                                <1>       ;DEC   SI                 ; CHECK FOR SET DASD TYPE 04
                                <1>       ; 11/04/2021
0000468A 4E                     <1>       dec    esi
0000468B 752A                   <1>       JNZ    short FS_ERR       ; BAD COMMAND EXIT IF NOT VALID TYPE
                                <1>
0000468D F687[85890100]04       <1>       TEST   byte [DSK_STATE+eDI], DRV_DET ; DRIVE DETERMINED ?
00004694 740B                   <1>       JZ     short ASSUME       ; IF STILL NOT DETERMINED ASSUME
00004696 B050                   <1>       MOV    AL,MED_DET+RATE_300
00004698 F687[85890100]02       <1>         TEST    byte [DSK_STATE+eDI], FMT_CAPA ; MULTIPLE FORMAT CAPABILITY ?
0000469F 7502                   <1>       JNZ    short OR_IT_IN            ; IF 1.2 M THEN DATA RATE 300
                                <1>
                                <1> ASSUME:
000046A1 B090                   <1>       MOV    AL,MED_DET+RATE_250 ; SET UP
                                <1>
                                <1> OR_IT_IN:
000046A3 0887[85890100]         <1>       OR     [DSK_STATE+eDI], AL ; OR IN THE CORRECT STATE
                                <1> S0:
000046A9 E847010000             <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
000046AE E86B060000             <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
                                <1>       ;POP   BX                 ; GET SAVED AL TO BL
                                <1>       ; 11/04/2021
000046B3 5B                     <1>       pop    ebx
000046B4 88D8                   <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
000046B6 C3                     <1>       RETn
                                <1>
                                <1> FS_ERR:
000046B7 C605[78890100]01       <1>       MOV    byte [DSKETTE_STATUS], BAD_CMD ; UNKNOWN STATE,BAD COMMAND
000046BE EBE9                   <1>       JMP    SHORT S0
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; SET_MEDIA  (AH = 18H)
                                <1> ;     THIS ROUTINE SETS THE TYPE OF MEDIA AND DATA RATE
                                <1> ;     TO BE USED FOR THE FOLLOWING FORMAT OPERATION.
                                <1> ;
                                <1> ; ON ENTRY:
                                <1> ;     [BP]   = SECTOR PER TRACK
                                <1> ;     [BP+1] = TRACK #
                                <1> ;     DI     = DRIVE #
                                <1> ;
                                <1> ; ON EXIT:
                                <1> ;     @DSKETTE_STATUS REFLECTS STATUS
                                <1> ;     IF NO ERROR:
                                <1> ;           AH = 0
                                <1> ;           CY = 0
                                <1> ;           ES = SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
                                <1> ;           DI/[BP+6] = OFFSET OF MEDIA/DRIVE PARAMETER TABLE
                                <1> ;     IF ERROR:
                                <1> ;           AH = @DSKETTE_STATUS
                                <1> ;           CY = 1
                                <1> ;--------------------------------------------------------------------------
                                <1> SET_MEDIA:
000046C0 E8FF000000             <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH.
000046C5 F687[85890100]01       <1>         TEST    byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR CHANGE LINE AVAILABLE
000046CC 7415                   <1>       JZ     short SM_CMOS      ; JUMP IF 40 TRACK DRIVE
000046CE E83A030000             <1>       CALL   MED_CHANGE         ; RESET CHANGE LINE
000046D3 803D[78890100]80       <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT TELL CALLER
000046DA 746B                   <1>       JE     short SM_RTN
000046DC C605[78890100]00       <1>       MOV    byte [DSKETTE_STATUS], 0 ; CLEAR STATUS
                                <1> SM_CMOS:
000046E3 E8FB060000             <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN (AL)
                                <1>       ;;20/02/2015
                                <1>       ;;JC   short MD_NOT_FND   ; ERROR IN CMOS
                                <1>       ;;OR   AL,AL              ; TEST FOR NO DRIVE
000046E8 745D                   <1>       JZ     short SM_RTN       ; RETURN IF SO
000046EA E863000000             <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
```

```
4084 000046EF 7231                <1>       JC     short MD_NOT_FND    ; TYPE NOT IN TABLE (BAD CMOS)
4085 000046F1 57                  <1>       PUSH   eDI                 ; SAVE REG.
4086 000046F2 31DB                <1>       XOR    eBX,eBX                   ; BX = INDEX TO DR. TYPE TABLE
4087 000046F4 B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
4088                              <1> DR_SEARCH:
4089 000046F9 8AA3[486C0000]      <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
4090 000046FF 80E47F              <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
4091 00004702 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH ?
4092 00004704 7516                <1>       JNE    short NXT_MD        ; NO, CHECK NEXT DRIVE TYPE
4093                              <1> DR_FND:
4094 00004706 8BBB[496C0000]      <1>       MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAM TABLE
4095                              <1> MD_SEARCH:
4096 0000470C 8A6704              <1>         MOV     AH, [eDI+MD.SEC_TRK]    ; GET SECTOR/TRACK
4097 0000470F 386500              <1>       CMP    [eBP],AH            ; MATCH?
4098 00004712 7508                <1>       JNE    short NXT_MD        ; NO, CHECK NEXT MEDIA
4099 00004714 8A670B              <1>         MOV     AH, [eDI+MD.MAX_TRK]    ; GET MAX. TRACK #
4100 00004717 386501              <1>       CMP    [eBP+1],AH          ; MATCH?
4101 0000471A 740F                <1>       JE     short MD_FND        ; YES, GO GET RATE
4102                              <1> NXT_MD:
4103                              <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
4104 0000471C 83C305              <1>         add ebx, 5 ; 18/02/2015
4105 0000471F E2D8                <1>       LOOP    DR_SEARCH
4106 00004721 5F                  <1>       POP    eDI                 ; RESTORE REG.
4107                              <1> MD_NOT_FND:
4108 00004722 C605[78890100]0C    <1>       MOV    byte [DSKETTE_STATUS], MED_NOT_FND ; ERROR, MEDIA TYPE NOT FOUND
4109 00004729 EB1C                <1>       JMP    SHORT SM_RTN        ; RETURN
4110                              <1> MD_FND:
4111 0000472B 8A470C              <1>         MOV     AL, [eDI+MD.RATE]       ; GET RATE
4112 0000472E 3C40                <1>       CMP    AL,RATE_300         ; DOUBLE STEP REQUIRED FOR RATE 300
4113 00004730 7502                <1>       JNE    short MD_SET
4114 00004732 0C20                <1>       OR     AL,DBL_STEP
4115                              <1> MD_SET:
4116                              <1>       ;MOV   [BP+6],DI           ; SAVE TABLE POINTER IN STACK
4117 00004734 897D0C              <1>       mov    [ebp+12], edi ; 18/02/2015
4118 00004737 0C10                <1>       OR     AL,MED_DET          ; SET MEDIA ESTABLISHED
4119 00004739 5F                  <1>       POP    eDI
4120 0000473A 80A7[85890100]0F    <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
4121 00004741 0887[85890100]      <1>       OR     [DSK_STATE+eDI], AL
4122                              <1>       ;MOV   AX, CS              ; SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
4123                              <1>       ;MOV   ES, AX              ; ES IS SEGMENT OF TABLE
4124                              <1> SM_RTN:
4125 00004747 E8A9000000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
4126 0000474C E8CD050000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
4127 00004751 C3                  <1>       RETn
4128                              <1>
4129                              <1> ;--------------------------------------------------------------
4130                              <1> ; DR_TYPE_CHECK                                          :
4131                              <1> ;     CHECK IF THE GIVEN DRIVE TYPE IN REGISTER (AL)             :
4132                              <1> ;     IS SUPPORTED IN BIOS DRIVE TYPE TABLE              :
4133                              <1> ; ON ENTRY:                                      :
4134                              <1> ;     AL = DRIVE TYPE                                  :
4135                              <1> ; ON EXIT:                                       :
4136                              <1> ;     CS = SEGMENT MEDIA/DRIVE PARAMETER TABLE (CODE)            :
4137                              <1> ;     CY = 0       DRIVE TYPE SUPPORTED                  :
4138                              <1> ;          BX = OFFSET TO MEDIA/DRIVE PARAMETER TABLE           :
4139                              <1> ;     CY = 1 DRIVE TYPE NOT SUPPORTED                :
4140                              <1> ; REGISTERS ALTERED: eBX                           :
4141                              <1> ;--------------------------------------------------------------
4142                              <1> DR_TYPE_CHECK:
4143                              <1>       ;PUSH  AX
4144                              <1>       ; 11/04/2021
4145 00004752 50                  <1>       push   eax
4146 00004753 51                  <1>       PUSH   eCX
4147 00004754 31DB                <1>       XOR    eBX,eBX                   ; BX = INDEX TO DR_TYPE TABLE
4148 00004756 B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
4149                              <1> TYPE_CHK:
4150 0000475B 8AA3[486C0000]      <1>       MOV    AH,[DR_TYPE+eBX]    ; GET DRIVE TYPE
4151 00004761 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
4152 00004763 740D                <1>       JE     short DR_TYPE_VALID ; YES, RETURN WITH CARRY RESET
4153                              <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
4154 00004765 83C305              <1>         add ebx, 5 ; 16/02/2015 (32 bit address modification)
4155 00004768 E2F1                <1>       LOOP    TYPE_CHK
4156                              <1>       ;
4157 0000476A BB[A76C0000]        <1>       mov    ebx, MD_TBL6        ; 1.44MB fd parameter table
4158                              <1>                                  ; Default for GET_PARM (11/12/2014)
4159                              <1>       ;
4160 0000476F F9                  <1>       STC                        ; DRIVE TYPE NOT FOUND IN TABLE
4161 00004770 EB06                <1>       JMP    SHORT TYPE_RTN
4162                              <1> DR_TYPE_VALID:
4163 00004772 8B9B[496C0000]      <1>       MOV    eBX,[DR_TYPE+eBX+1]       ; BX = MEDIA TABLE
4164                              <1> TYPE_RTN:
4165 00004778 59                  <1>       POP    eCX
4166                              <1>       ;POP   AX
4167                              <1>       ; 11/04/2021
4168 00004779 58                  <1>       pop    eax
4169 0000477A C3                  <1>       RETn
4170                              <1>
4171                              <1> ;--------------------------------------------------------------
4172                              <1> ; SEND_SPEC                                          :
4173                              <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
4174                              <1> ;     THE DRIVE PARAMETER TABLE POINTED BY @DISK_POINTER   :
4175                              <1> ; ON ENTRY:  @DISK_POINTER = DRIVE PARAMETER TABLE         :
4176                              <1> ; ON EXIT:   NONE                                  :
4177                              <1> ; REGISTERS ALTERED: CX, DX                            :
4178                              <1> ;--------------------------------------------------------------
4179                              <1> SEND_SPEC:
4180 0000477B 50                  <1>       PUSH   eAX                 ; SAVE AX
4181 0000477C B8[A2470000]        <1>       MOV    eAX, SPECBAC        ; LOAD ERROR ADDRESS
4182 00004781 50                  <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
4183 00004782 B403                <1>       MOV    AH,03H              ; SPECIFY COMMAND
4184 00004784 E869070000          <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
4185 00004789 28D2                <1>       SUB    DL,DL               ; FIRST SPECIFY BYTE
4186 0000478B E85C060000          <1>       CALL   GET_PARM            ; GET PARAMETER TO AH
4187 00004790 E85D070000          <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
4188 00004795 B201                <1>       MOV    DL,1                ; SECOND SPECIFY BYTE
```

```
4189 00004797 E850060000          <1>       CALL   GET_PARM           ; GET PARAMETER TO AH
4190 0000479C E851070000          <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
4191 000047A1 58                  <1>       POP    eAX                ; POP ERROR RETURN
4192                              <1> SPECBAC:
4193 000047A2 58                  <1>       POP    eAX                ; RESTORE ORIGINAL AX VALUE
4194 000047A3 C3                  <1>       RETn
4195                              <1>
4196                              <1> ;--------------------------------------------------------------
4197                              <1> ; SEND_SPEC_MD                                           :
4198                              <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
4199                              <1> ;     THE MEDIA/DRIVE PARAMETER TABLE POINTED BY (CS:BX)   :
4200                              <1> ; ON ENTRY:  CS:BX = MEDIA/DRIVE PARAMETER TABLE         :
4201                              <1> ; ON EXIT:   NONE                                  :
4202                              <1> ; REGISTERS ALTERED: AX                                  :
4203                              <1> ;--------------------------------------------------------------
4204                              <1> SEND_SPEC_MD:
4205 000047A4 50                  <1>       PUSH   eAX                ; SAVE RATE DATA
4206 000047A5 B8[C2470000]        <1>       MOV    eAX, SPEC_ESBAC          ; LOAD ERROR ADDRESS
4207 000047AA 50                  <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
4208 000047AB B403                <1>       MOV    AH,03H             ; SPECIFY COMMAND
4209 000047AD E840070000          <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
4210 000047B2 8A23                <1>         MOV     AH, [eBX+MD.SPEC1]      ; GET 1ST SPECIFY BYTE
4211 000047B4 E839070000          <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
4212 000047B9 8A6301              <1>         MOV     AH, [eBX+MD.SPEC2]      ; GET SECOND SPECIFY BYTE
4213 000047BC E831070000          <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
4214 000047C1 58                  <1>       POP    eAX                ; POP ERROR RETURN
4215                              <1> SPEC_ESBAC:
4216 000047C2 58                  <1>       POP    eAX                ; RESTORE ORIGINAL AX VALUE
4217 000047C3 C3                  <1>       RETn
4218                              <1>
4219                              <1> ;-----------------------------------------------------------------------------
4220                              <1> ; XLAT_NEW
4221                              <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM COMPATIBLE
4222                              <1> ;     MODE TO NEW ARCHITECTURE.
4223                              <1> ;
4224                              <1> ; ON ENTRY:  DI = DRIVE #
4225                              <1> ;-----------------------------------------------------------------------------
4226                              <1> XLAT_NEW:
4227 000047C4 83FF01              <1>       CMP    eDI,1                    ; VALID DRIVE
4228 000047C7 7725                <1>       JA     short XN_OUT             ; IF INVALID BACK
4229 000047C9 80BF[85890100]00    <1>       CMP    byte [DSK_STATE+eDI], 0          ; NO DRIVE ?
4230 000047D0 741D                <1>       JZ     short DO_DET             ; IF NO DRIVE ATTEMPT DETERMINE
4231 000047D2 6689F9              <1>       MOV    CX,DI                    ; CX = DRIVE NUMBER
4232 000047D5 C0E102              <1>       SHL    CL,2                     ; CL = SHIFT COUNT, A=0, B=4
4233 000047D8 A0[84890100]        <1>       MOV    AL, [HF_CNTRL]                ; DRIVE INFORMATION
4234 000047DD D2C8                <1>       ROR    AL,CL                    ; TO LOW NIBBLE
4235 000047DF 2407                <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA     ; KEEP DRIVE BITS
4236 000047E1 80A7[85890100]F8    <1>         AND     byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA)
4237 000047E8 0887[85890100]      <1>       OR     [DSK_STATE+eDI], AL       ; UPDATE DRIVE STATE
4238                              <1> XN_OUT:
4239 000047EE C3                  <1>       RETn
4240                              <1> DO_DET:
4241 000047EF E8A1080000          <1>       CALL   DRIVE_DET                ; TRY TO DETERMINE
4242 000047F4 C3                  <1>       RETn
4243                              <1>
4244                              <1> ;-----------------------------------------------------------------------------
4245                              <1> ; XLAT_OLD
4246                              <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM NEW
4247                              <1> ;     ARCHITECTURE TO COMPATIBLE MODE.
4248                              <1> ;
4249                              <1> ; ON ENTRY:  DI = DRIVE
4250                              <1> ;-----------------------------------------------------------------------------
4251                              <1> XLAT_OLD:
4252 000047F5 83FF01              <1>       CMP    eDI,1              ; VALID DRIVE ?
4253                              <1>         ;JA     short XO_OUT            ; IF INVALID BACK
4254 000047F8 0F8786000000        <1>         ja      XO_OUT
4255 000047FE 80BF[85890100]00    <1>         CMP  byte [DSK_STATE+eDI],0    ; NO DRIVE ?
4256 00004805 747D                <1>       JZ     short XO_OUT       ; IF NO DRIVE TRANSLATE DONE
4257                              <1>
4258                              <1> ;-----      TEST FOR SAVED DRIVE INFORMATION ALREADY SET
4259                              <1>
4260 00004807 6689F9              <1>       MOV    CX,DI              ; CX = DRIVE NUMBER
4261 0000480A C0E102              <1>       SHL    CL,2               ; CL = SHIFT COUNT, A=0, B=4
4262 0000480D B402                <1>       MOV    AH,FMT_CAPA        ; LOAD MULTIPLE DATA RATE BIT MASK
4263 0000480F D2CC                <1>       ROR    AH,CL              ; ROTATE BY MASK
4264 00004811 8425[84890100]      <1>       TEST   [HF_CNTRL], AH           ; MULTIPLE-DATA RATE DETERMINED ?
4265 00004817 751C                <1>       JNZ    short SAVE_SET           ; IF SO, NO NEED TO RE-SAVE
4266                              <1>
4267                              <1> ;-----      ERASE DRIVE BITS IN @HF_CNTRL FOR THIS DRIVE
4268                              <1>
4269 00004819 B407                <1>       MOV    AH,DRV_DET+FMT_CAPA+TRK_CAPA ; MASK TO KEEP
4270 0000481B D2CC                <1>       ROR    AH,CL              ; FIX MASK TO KEEP
4271 0000481D F6D4                <1>       NOT    AH                 ; TRANSLATE MASK
4272 0000481F 2025[84890100]      <1>       AND    [HF_CNTRL], AH           ; KEEP BITS FROM OTHER DRIVE INTACT
4273                              <1>
4274                              <1> ;-----      ACCESS CURRENT DRIVE BITS AND STORE IN @HF_CNTRL
4275                              <1>
4276 00004825 8A87[85890100]      <1>       MOV    AL, [DSK_STATE+eDI] ; ACCESS STATE
4277 0000482B 2407                <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
4278 0000482D D2C8                <1>       ROR    AL,CL              ; FIX FOR THIS DRIVE
4279 0000482F 0805[84890100]      <1>       OR     [HF_CNTRL], AL           ; UPDATE SAVED DRIVE STATE
4280                              <1>
4281                              <1> ;-----      TRANSLATE TO COMPATIBILITY MODE
4282                              <1>
4283                              <1> SAVE_SET:
4284 00004835 8AA7[85890100]      <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
4285 0000483B 88E7                <1>       MOV    BH,AH              ; TO BH FOR LATER
4286 0000483D 80E4C0              <1>       AND    AH,RATE_MSK        ; KEEP ONLY RATE
4287 00004840 80FC00              <1>       CMP    AH,RATE_500        ; RATE 500 ?
4288 00004843 7410                <1>       JZ     short CHK_144      ; YES 1.2/1.2 OR 1.44/1.44
4289 00004845 B001                <1>       MOV    AL,M3D1U           ; AL = 360 IN 1.2 UNESTABLISHED
4290 00004847 80FC40              <1>       CMP    AH,RATE_300        ; RATE 300 ?
4291 0000484A 7518                <1>       JNZ    short CHK_250      ; NO, 360/360, 720/720 OR 720/1.44
4292 0000484C F6C720              <1>       TEST   BH,DBL_STEP        ; CHECK FOR DOUBLE STEP
4293 0000484F 751F                <1>       JNZ    short TST_DET      ; MUST BE 360 IN 1.2
```

```
                                <1> UNKNO:
00004851 B007                   <1>       MOV    AL,MED_UNK          ; NONE OF THE ABOVE
00004853 EB22                   <1>       JMP    SHORT AL_SET        ; PROCESS COMPLETE
                                <1> CHK_144:
00004855 E889050000             <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
                                <1>       ;;20/02/2015
                                <1>       ;;JC   short UNKNO         ; ERROR, SET 'NONE OF ABOVE'
0000485A 74F5                   <1>       jz     short UNKNO ;; 20/02/2015
0000485C 3C02                   <1>       CMP    AL,2                ; 1.2MB DRIVE ?
0000485E 75F1                   <1>       JNE    short UNKNO         ; NO, GO SET 'NONE OF ABOVE'
00004860 B002                   <1>       MOV    AL,M1D1U            ; AL = 1.2 IN 1.2 UNESTABLISHED
00004862 EB0C                   <1>       JMP    SHORT TST_DET
                                <1> CHK_250:
00004864 B000                   <1>       MOV    AL,M3D3U            ; AL = 360 IN 360 UNESTABLISHED
00004866 80FC80                 <1>       CMP    AH,RATE_250         ; RATE 250 ?
00004869 75E6                   <1>       JNZ    short UNKNO         ; IF SO FALL IHRU
0000486B F6C701                 <1>       TEST   BH,TRK_CAPA         ; 80 TRACK CAPABILITY ?
0000486E 75E1                   <1>       JNZ    short UNKNO         ; IF SO JUMP, FALL THRU TEST DET
                                <1> TST_DET:
00004870 F6C710                 <1>       TEST   BH,MED_DET          ; DETERMINED ?
00004873 7402                   <1>       JZ     short AL_SET        ; IF NOT THEN SET
00004875 0403                   <1>       ADD    AL,3                ; MAKE DETERMINED/ESTABLISHED
                                <1> AL_SET:
00004877 80A7[85890100]F8       <1>       AND    byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA) ; CLEAR DRIVE
0000487E 0887[85890100]         <1>       OR     [DSK_STATE+eDI], AL ; REPLACE WITH COMPATIBLE MODE
                                <1> XO_OUT:
00004884 C3                     <1>       RETn
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; RD_WR_VF
                                <1> ;     COMMON READ, WRITE AND VERIFY:
                                <1> ;     MAIN LOOP FOR STATE RETRIES.
                                <1> ;
                                <1> ; ON ENTRY:  AH = READ/WRITE/VERIFY NEC PARAMETER
                                <1> ;            AL = READ/WRITE/VERIFY DMA PARAMETER
                                <1> ;
                                <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                                <1> ;--------------------------------------------------------------------------
                                <1> RD_WR_VF:
                                <1>       ; 11/04/2021 (32 bit push/pop, AX -> eAX)
00004885 50                     <1>       PUSH   eAX                 ; SAVE DMA, NEC PARAMETERS
00004886 E839FFFFFF             <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
0000488B E8E8000000             <1>       CALL   SETUP_STATE         ; INITIALIZE START AND END RATE
00004890 58                     <1>       POP    eAX                 ; RESTORE READ/WRITE/VERIFY
                                <1> DO_AGAIN:
00004891 50                     <1>       PUSH   eAX                 ; SAVE READ/WRITE/VERIFY PARAMETER
00004892 E876010000             <1>       CALL   MED_CHANGE          ; MEDIA CHANGE AND RESET IF CHANGED
00004897 58                     <1>       POP    eAX                 ; RESTORE READ/WRITE/VERIFY
00004898 0F82C3000000           <1>         JC     RWV_END                ; MEDIA CHANGE ERROR OR TIME-OUT
                                <1> RWV:
0000489E 50                     <1>       PUSH   eAX                 ; SAVE READ/WRITE/VERIFY PARAMETER
0000489F 8AB7[85890100]         <1>       MOV    DH, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
000048A5 80E6C0                 <1>       AND    DH,RATE_MSK         ; KEEP ONLY RATE
000048A8 E836050000             <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL (AL)
                                <1>       ;;20/02/2015
                                <1>       ;;JC   short RWV_ASSUME    ; ERROR IN CMOS
000048AD 7451                   <1>       jz     short RWV_ASSUME ; 20/02/2015
000048AF 3C01                   <1>       CMP    AL,1                ; 40 TRACK DRIVE?
000048B1 750D                   <1>       JNE    short RWV_1         ; NO, BYPASS CMOS VALIDITY CHECK
000048B3 F687[85890100]01       <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR 40 TRACK DRIVE
000048BA 7413                   <1>       JZ     short RWV_2         ; YES, CMOS IS CORRECT
000048BC B002                   <1>       MOV    AL,2                ; CHANGE TO 1.2M
000048BE EB0F                   <1>       JMP    SHORT RWV_2
                                <1> RWV_1:
000048C0 720D                   <1>       JB     short RWV_2         ; NO DRIVE SPECIFIED, CONTINUE
000048C2 F687[85890100]01       <1>       TEST    byte [DSK_STATE+eDI], TRK_CAPA ; IS IT REALLY 40 TRACK?
000048C9 7504                   <1>       JNZ    short RWV_2         ; NO, 80 TRACK
000048CB B001                   <1>       MOV    AL,1                ; IT IS 40 TRACK, FIX CMOS VALUE
000048CD EB04                   <1>       jmp    short rwv_3
                                <1> RWV_2:
000048CF 08C0                   <1>       OR     AL,AL               ; TEST FOR NO DRIVE
000048D1 742D                   <1>       JZ     short RWV_ASSUME    ; ASSUME TYPE, USE MAX TRACK
                                <1> rwv_3:
000048D3 E87AFEFFFF             <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL.
000048D8 7226                   <1>       JC     short RWV_ASSUME    ; TYPE NOT IN TABLE (BAD CMOS)
                                <1>
                                <1> ;-----       SEARCH FOR MEDIA/DRIVE PARAMETER TABLE
                                <1>
000048DA 57                     <1>       PUSH   eDI                 ; SAVE DRIVE #
000048DB 31DB                   <1>       XOR    eBX,eBX                  ; BX = INDEX TO DR_TYPE TABLE
000048DD B906000000             <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
                                <1> RWV_DR_SEARCH:
000048E2 8AA3[486C0000]         <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
000048E8 80E47F                 <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
000048EB 38E0                   <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
000048ED 750B                   <1>       JNE    short RWV_NXT_MD    ; NO, CHECK NEXT DRIVE TYPE
                                <1> RWV_DR_FND:
000048EF 8BBB[496C0000]         <1>       MOV    eDI, [DR_TYPE+eBX+1]       ; DI = MEDIA/DRIVE PARAMETER TABLE
                                <1> RWV_MD_SEARH:
000048F5 3A770C                 <1>         CMP     DH, [eDI+MD.RATE]        ; MATCH?
000048F8 741B                   <1>       JE     short RWV_MD_FND    ; YES, GO GET 1ST SPECIFY BYTE
                                <1> RWV_NXT_MD:
                                <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
000048FA 83C305                 <1>       add    eBX, 5
000048FD E2E3                   <1>       LOOP   RWV_DR_SEARCH
000048FF 5F                     <1>       POP    eDI                 ; RESTORE DRIVE #
                                <1>
                                <1> ;-----       ASSUME PRIMARY DRIVE IS INSTALLED AS SHIPPED
                                <1>
                                <1> RWV_ASSUME:
00004900 BB[666C0000]           <1>       MOV    eBX, MD_TBL1        ; POINT TO 40 TRACK 250 KBS
00004905 F687[85890100]01       <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; TEST FOR 80 TRACK
0000490C 740A                   <1>       JZ     short RWV_MD_FND1   ; MUST BE 40 TRACK
0000490E BB[806C0000]           <1>       MOV    eBX, MD_TBL3        ; POINT TO 80 TRACK 500 KBS
00004913 EB03                   <1>       JMP    short RWV_MD_FND1   ; GO SPECIFY PARAMTERS
```

```
                                <1>
                                <1> ;-----      CS:BX POINTS TO MEDIA/DRIVE PARAMETER TABLE
                                <1>
                                <1> RWV_MD_FND:
00004915 89FB                   <1>       MOV   eBX,eDI                   ; BX = MEDIA/DRIVE PARAMETER TABLE
00004917 5F                     <1>       POP   eDI                 ; RESTORE DRIVE #
                                <1>
                                <1> ;-----      SEND THE SPECIFY COMMAND TO THE CONTROLLER
                                <1>
                                <1> RWV_MD_FND1:
00004918 E887FEFFFF             <1>       CALL  SEND_SPEC_MD
0000491D E85B010000             <1>       CALL  CHK_LASTRATE        ; ZF=1 ATTEMP RATE IS SAME AS LAST RATE
00004922 7405                   <1>       JZ    short RWV_DBL       ; YES,SKIP SEND RATE COMMAND
00004924 E834010000             <1>       CALL  SEND_RATE           ; SEND DATA RATE TO NEC
                                <1> RWV_DBL:
00004929 53                     <1>       PUSH  eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
0000492A E80F040000             <1>       CALL  SETUP_DBL           ; CHECK FOR DOUBLE STEP
0000492F 5B                     <1>       POP   eBX                 ; RESTORE ADDRESS
00004930 7222                   <1>       JC    short CHK_RET       ; ERROR FROM READ ID, POSSIBLE RETRY
00004932 58                     <1>       POP   eAX                 ; RESTORE NEC, DMA COMMAND
00004933 50                     <1>       PUSH  eAX                 ; SAVE NEC COMMAND
00004934 53                     <1>       PUSH  eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
00004935 E858010000             <1>       CALL  DMA_SETUP           ; SET UP THE DMA
0000493A 5B                     <1>       POP   eBX
0000493B 58                     <1>       POP   eAX                 ; RESTORE NEC COMMAND
0000493C 722D                   <1>       JC    short RWV_BAC       ; CHECK FOR DMA BOUNDARY ERROR
0000493E 50                     <1>       PUSH  eAX                 ; SAVE NEC COMMAND
0000493F 53                     <1>       PUSH  eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
00004940 E831020000             <1>       CALL  NEC_INIT            ; INITIALIZE NEC
00004945 5B                     <1>       POP   eBX                 ; RESTORE ADDRESS
00004946 720C                   <1>       JC    short CHK_RET       ; ERROR - EXIT
00004948 E859020000             <1>       CALL  RWV_COM                   ; OP CODE COMMON TO READ/WRITE/VERIFY
0000494D 7205                   <1>       JC    short CHK_RET       ; ERROR - EXIT
0000494F E89E020000             <1>       CALL  NEC_TERM            ; TERMINATE, GET STATUS, ETC.
                                <1> CHK_RET:
00004954 E83D030000             <1>       CALL  RETRY               ; CHECK FOR, SETUP RETRY
00004959 58                     <1>       POP   eAX                 ; RESTORE READ/WRITE/VERIFY PARAMETER
0000495A 7305                   <1>       JNC   short RWV_END       ; CY = 0 NO RETRY
0000495C E930FFFFFF             <1>         JMP     DO_AGAIN              ; CY = 1 MEANS RETRY
                                <1> RWV_END:
00004961 E8E8020000             <1>       CALL  DSTATE              ; ESTABLISH STATE IF SUCCESSFUL
00004966 E87B030000             <1>       CALL  NUM_TRANS           ; AL = NUMBER TRANSFERRED
                                <1> RWV_BAC:                        ; BAD DMA ERROR ENTRY
0000496B 50                     <1>       PUSH  eAX                 ; SAVE NUMBER TRANSFERRED
0000496C E884FEFFFF             <1>       CALL  XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
00004971 58                     <1>       POP   eAX                 ; RESTORE NUMBER TRANSFERRED
00004972 E8A7030000             <1>       CALL  SETUP_END           ; VARIOUS CLEANUPS
00004977 C3                     <1>       RETn
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; SETUP_STATE:     INITIALIZES START AND END RATES.
                                <1> ;--------------------------------------------------------------------------
                                <1> SETUP_STATE:
00004978 F687[85890100]10       <1>       TEST  byte [DSK_STATE+eDI], MED_DET ; MEDIA DETERMINED ?
0000497F 7537                   <1>       JNZ   short J1C           ; NO STATES IF DETERMINED
00004981 66B84000               <1>         MOV     AX,(RATE_500*256)+RATE_300  ; AH = START RATE, AL = END RATE
00004985 F687[85890100]04       <1>       TEST  byte [DSK_STATE+eDI],DRV_DET ; DRIVE ?
0000498C 740D                   <1>       JZ    short AX_SET        ; DO NOT KNOW DRIVE
0000498E F687[85890100]02       <1>       TEST  byte [DSK_STATE+eDI], FMT_CAPA ; MULTI-RATE?
00004995 7504                   <1>       JNZ   short AX_SET        ; JUMP IF YES
00004997 66B88080               <1>         MOV     AX,RATE_250*257         ; START A END RATE 250 FOR 360 DRIVE
                                <1> AX_SET:
0000499B 80A7[85890100]1F       <1>       AND   byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP) ; TURN OFF THE RATE
000049A2 08A7[85890100]         <1>       OR    [DSK_STATE+eDI], AH ; RATE FIRST TO TRY
000049A8 8025[80890100]F3       <1>       AND   byte [LASTRATE], ~STRT_MSK ; ERASE LAST TO TRY RATE BITS
000049AF C0C804                 <1>       ROR   AL,4                ; TO OPERATION LAST RATE LOCATION
000049B2 0805[80890100]         <1>       OR    [LASTRATE], AL            ; LAST RATE
                                <1> J1C:
000049B8 C3                     <1>       RETn
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ;  FMT_INIT: ESTABLISH STATE IF UNESTABLISHED AT FORMAT TIME.
                                <1> ;--------------------------------------------------------------------------
                                <1> FMT_INIT:
000049B9 F687[85890100]10       <1>       TEST  byte [DSK_STATE+eDI], MED_DET ; IS MEDIA ESTABLISHED
000049C0 7546                   <1>       JNZ   short F1_OUT        ; IF SO RETURN
000049C2 E81C040000             <1>       CALL  CMOS_TYPE           ; RETURN DRIVE TYPE IN AL
                                <1>       ;; 20/02/2015
                                <1>       ;;JC  short CL_DRV        ; ERROR IN CMOS ASSUME NO DRIVE
000049C7 7440                   <1>       jz    short CL_DRV ;; 20/02/2015
000049C9 FEC8                   <1>       DEC   AL                  ; MAKE ZERO ORIGIN
                                <1>       ;;JS  short CL_DRV        ; NO DRIVE IF AL 0
000049CB 8AA7[85890100]         <1>       MOV   AH, [DSK_STATE+eDI] ; AH = CURRENT STATE
000049D1 80E40F                 <1>       AND   AH, ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR
000049D4 08C0                   <1>       OR    AL,AL               ; CHECK FOR 360
000049D6 7505                   <1>       JNZ   short N_360         ; IF 360 WILL BE 0
000049D8 80CC90                 <1>       OR    AH,MED_DET+RATE_250 ; ESTABLISH MEDIA
000049DB EB25                   <1>       JMP   SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
                                <1> N_360:
000049DD FEC8                   <1>       DEC   AL                  ; 1.2 M DRIVE
000049DF 7505                   <1>       JNZ   short N_12          ; JUMP IF NOT
                                <1> F1_RATE:
000049E1 80CC10                 <1>       OR    AH,MED_DET+RATE_500 ; SET FORMAT RATE
000049E4 EB1C                   <1>       JMP   SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
                                <1> N_12:
000049E6 FEC8                   <1>       DEC   AL                  ; CHECK FOR TYPE 3
000049E8 750F                   <1>       JNZ   short N_720         ; JUMP IF NOT
000049EA F6C404                 <1>       TEST  AH,DRV_DET          ; IS DRIVE DETERMINED
000049ED 7410                   <1>       JZ    short ISNT_12       ; TREAT AS NON 1.2 DRIVE
000049EF F6C402                 <1>       TEST  AH,FMT_CAPA         ; IS 1.2M
000049F2 740B                   <1>       JZ    short ISNT_12       ; JUMP IF NOT
000049F4 80CC50                 <1>       OR    AH,MED_DET+RATE_300 ; RATE 300
000049F7 EB09                   <1>       JMP   SHORT SKP_STATE           ; CONTINUE
                                <1> N_720:
000049F9 FEC8                   <1>       DEC   AL                  ; CHECK FOR TYPE 4
```

```
4504 000049FB 750C              <1>       JNZ    short CL_DRV        ; NO DRIVE, CMOS BAD
4505 000049FD EBE2              <1>       JMP    SHORT F1_RATE
4506                            <1> ISNT_12:
4507 000049FF 80CC90            <1>       OR     AH,MED_DET+RATE_250 ; MUST BE RATE 250
4508                            <1>
4509                            <1> SKP_STATE:
4510 00004A02 88A7[85890100]    <1>       MOV    [DSK_STATE+eDI], AH ; STORE AWAY
4511                            <1> F1_OUT:
4512 00004A08 C3                <1>       RETn
4513                            <1> CL_DRV:
4514 00004A09 30E4              <1>       XOR    AH,AH               ; CLEAR STATE
4515 00004A0B EBF5              <1>       JMP    SHORT SKP_STATE           ; SAVE IT
4516                            <1>
4517                            <1> ;--------------------------------------------------------------------------
4518                            <1> ; MED_CHANGE
4519                            <1> ;     CHECKS FOR MEDIA CHANGE, RESETS MEDIA CHANGE,
4520                            <1> ;     CHECKS MEDIA CHANGE AGAIN.
4521                            <1> ;
4522                            <1> ; ON EXIT:  CY = 1 MEANS MEDIA CHANGE OR TIMEOUT
4523                            <1> ;           @DSKETTE_STATUS = ERROR CODE
4524                            <1> ;--------------------------------------------------------------------------
4525                            <1> MED_CHANGE:
4526 00004A0D E876060000        <1>       CALL   READ_DSKCHNG        ; READ DISK CHANCE LINE STATE
4527 00004A12 7447              <1>       JZ     short MC_OUT        ; BYPASS HANDLING DISK CHANGE LINE
4528 00004A14 80A7[85890100]EF  <1>       AND    byte [DSK_STATE+eDI], ~MED_DET ; CLEAR STATE FOR THIS DRIVE
4529                            <1>
4530                            <1> ;     THIS SEQUENCE ENSURES WHENEVER A DISKETTE IS CHANGED THAT
4531                            <1> ;     ON THE NEXT OPERATION THE REQUIRED MOTOR START UP TIME WILL
4532                            <1> ;     BE WAITED. (DRIVE MOTOR MAY GO OFF UPON DOOR OPENING).
4533                            <1>
4534 00004A1B 6689F9            <1>       MOV    CX,DI               ; CL = DRIVE 0
4535 00004A1E B001              <1>       MOV    AL,1                ; MOTOR ON BIT MASK
4536 00004A20 D2E0              <1>       SHL    AL,CL               ; TO APPROPRIATE POSITION
4537 00004A22 F6D0              <1>       NOT    AL                  ; KEEP ALL BUT MOTOR ON
4538 00004A24 FA                <1>       CLI                        ; NO INTERRUPTS
4539 00004A25 2005[76890100]    <1>       AND    [MOTOR_STATUS], AL  ; TURN MOTOR OFF INDICATOR
4540 00004A2B FB                <1>       STI                        ; INTERRUPTS ENABLED
4541 00004A2C E800040000        <1>       CALL   MOTOR_ON            ; TURN MOTOR ON
4542                            <1>
4543                            <1> ;-----      THIS SEQUENCE OF SEEKS IS USED TO RESET DISKETTE CHANGE SIGNAL
4544                            <1>
4545 00004A31 E876F9FFFF        <1>       CALL   DSK_RESET           ; RESET NEC
4546 00004A36 B501              <1>       MOV    CH,01H              ; MOVE TO CYLINDER 1
4547 00004A38 E8EF040000        <1>       CALL   SEEK                ; ISSUE SEEK
4548 00004A3D 30ED              <1>       XOR    CH,CH               ; MOVE TO CYLINDER 0
4549 00004A3F E8E8040000        <1>       CALL   SEEK                ; ISSUE SEEK
4550 00004A44 C605[78890100]06  <1>       MOV    byte [DSKETTE_STATUS], MEDIA_CHANGE ; STORE IN STATUS
4551                            <1> OK1:
4552 00004A4B E838060000        <1>       CALL   READ_DSKCHNG        ; CHECK MEDIA CHANGED AGAIN
4553 00004A50 7407              <1>       JZ     short OK2           ; IF ACTIVE, NO DISKETTE, TIMEOUT
4554                            <1> OK4:
4555 00004A52 C605[78890100]80  <1>       MOV    byte [DSKETTE_STATUS], TIME_OUT ; TIMEOUT IF DRIVE EMPTY
4556                            <1> OK2:
4557 00004A59 F9                <1>       STC                        ; MEDIA CHANGED, SET CY
4558 00004A5A C3                <1>       RETn
4559                            <1> MC_OUT:
4560 00004A5B F8                <1>       CLC                        ; NO MEDIA CHANGED, CLEAR CY
4561 00004A5C C3                <1>       RETn
4562                            <1>
4563                            <1> ;--------------------------------------------------------------------------
4564                            <1> ; SEND_RATE
4565                            <1> ;     SENDS DATA RATE COMMAND TO NEC
4566                            <1> ; ON ENTRY:  DI = DRIVE #
4567                            <1> ; ON EXIT:   NONE
4568                            <1> ; REGISTERS ALTERED: DX
4569                            <1> ;--------------------------------------------------------------------------
4570                            <1> SEND_RATE:
4571                            <1>       ;PUSH  AX                  ; SAVE REG.
4572                            <1>       ; 11/04/2021
4573 00004A5D 50                <1>       push   eax
4574 00004A5E 8025[80890100]3F  <1>       AND    byte [LASTRATE], ~SEND_MSK ; ELSE CLEAR LAST RATE ATTEMPTED
4575 00004A65 8A87[85890100]    <1>       MOV    AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
4576 00004A6B 24C0              <1>       AND    AL,SEND_MSK         ; KEEP ONLY RATE BITS
4577 00004A6D 0805[80890100]    <1>       OR     [LASTRATE], AL            ; SAVE NEW RATE FOR NEXT CHECK
4578 00004A73 C0C002            <1>       ROL    AL,2                ; MOVE TO BIT OUTPUT POSITIONS
4579 00004A76 66BAF703          <1>       MOV    DX,03F7H            ; OUTPUT NEW DATA RATE
4580 00004A7A EE                <1>       OUT    DX,AL
4581                            <1>       ;POP   AX                  ; RESTORE REG.
4582                            <1>       ; 11/04/2021
4583 00004A7B 58                <1>       pop    eax
4584 00004A7C C3                <1>       RETn
4585                            <1>
4586                            <1> ;--------------------------------------------------------------------------
4587                            <1> ; CHK_LASTRATE
4588                            <1> ;     CHECK PREVIOUS DATE RATE SNT TO THE CONTROLLER.
4589                            <1> ; ON ENTRY:
4590                            <1> ;     DI = DRIVE #
4591                            <1> ; ON EXIT:
4592                            <1> ;     ZF =  1 DATA RATE IS THE SAME AS THE LAST RATE SENT TO NEC
4593                            <1> ;     ZF =  0 DATA RATE IS DIFFERENT FROM LAST RATE
4594                            <1> ; REGISTERS ALTERED: DX
4595                            <1> ;--------------------------------------------------------------------------
4596                            <1> CHK_LASTRATE:
4597                            <1>       ;PUSH  AX                  ; SAVE REG.
4598                            <1>       ; 11/04/2021
4599 00004A7D 50                <1>       push   eax
4600 00004A7E 2225[80890100]    <1>       AND    AH, [LASTRATE]            ; GET LAST DATA RATE SELECTED
4601 00004A84 8A87[85890100]    <1>       MOV    AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
4602 00004A8A 6625C0C0          <1>         AND     AX, SEND_MSK*257        ; KEEP ONLY RATE BITS OF BOTH
4603 00004A8E 38E0              <1>       CMP    AL, AH              ; COMPARE TO PREVIOUSLY TRIED
4604                            <1>                                  ; ZF = 1 RATE IS THE SAME
4605                            <1>       ;POP   AX                  ; RESTORE REG.
4606                            <1>       ; 11/04/2021
4607 00004A90 58                <1>       pop    eax
4608 00004A91 C3                <1>       RETn
```

```
                                <1>
                                <1> ;-----------------------------------------------------------------------------
                                <1> ; DMA_SETUP
                                <1> ;     THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS.
                                <1> ;
                                <1> ; ON ENTRY:  AL = DMA COMMAND
                                <1> ;
                                <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                                <1> ;-----------------------------------------------------------------------------
                                <1>
                                <1> ; SI = Head #, # of Sectors or DASD Type
                                <1>
                                <1> ; 22/08/2015
                                <1> ; 08/02/2015 - Protected Mode Modification
                                <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
00004A92 8B5504                 <1>         mov     edx, [ebp+4]        ; Buffer address
00004A95 F7C2000000FF           <1>         test    edx, 0FF000000h             ; 16 MB limit (22/08/2015, bugfix)
00004A9B 756C                   <1>         jnz     short dma_bnd_err_stc
                                <1>         ;
                                <1>         ;push  ax                  ; DMA command
                                <1>         ; 11/04/2021
00004A9D 50                     <1>         push    eax
00004A9E 52                     <1>         push    edx                 ; *
00004A9F B203                   <1>         mov     dl, 3               ; GET BYTES/SECTOR PARAMETER
00004AA1 E846030000             <1>         call    GET_PARM            ;
00004AA6 88E1                   <1>         mov     cl, ah                      ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
00004AA8 6689F0                 <1>         mov     ax, si              ; Sector count
00004AAB 88C4                   <1>         mov     ah, al              ; AH =  # OF SECTORS
00004AAD 28C0                   <1>         sub     al, al              ; AL = 0, AX = # SECTORS * 256
00004AAF 66D1E8                 <1>         shr     ax, 1               ; AX = # SECTORS * 128
00004AB2 66D3E0                 <1>         shl     ax, cl              ; SHIFT BY PARAMETER VALUE
00004AB5 6648                   <1>         dec     ax                  ; -1 FOR DMA VALUE
00004AB7 6689C1                 <1>         mov     cx, ax
00004ABA 5A                     <1>         pop     edx                 ; *
                                <1>         ;pop   ax
                                <1>         ; 11/04/2021
00004ABB 58                     <1>         pop     eax
00004ABC 3C42                   <1>         cmp     al, 42h
00004ABE 7507                   <1>           jne     short NOT_VERF
00004AC0 BA0000FF00             <1>         mov     edx, 0FF0000h
00004AC5 EB08                   <1>         jmp     short J33
                                <1> NOT_VERF:
00004AC7 6601CA                 <1>         add     dx, cx              ; check for overflow
00004ACA 723E                   <1>         jc      short dma_bnd_err
                                <1>         ;
00004ACC 6629CA                 <1>         sub     dx, cx              ; Restore start address
                                <1> J33:
00004ACF FA                     <1>         CLI                         ; DISABLE INTERRUPTS DURING DMA SET-UP
00004AD0 E60C                   <1>         OUT     DMA+12,AL           ; SET THE FIRST/LA5T F/F
                                <1>         IODELAY                             ; WAIT FOR I/O
2968 00004AD2 EB00              <2>  jmp short $+2
2969 00004AD4 EB00              <2>  jmp short $+2
00004AD6 E60B                   <1>         OUT     DMA+11,AL           ; OUTPUT THE MODE BYTE
00004AD8 89D0                   <1>         mov     eax, edx            ; Buffer address
00004ADA E604                   <1>         OUT     DMA+4,AL            ; OUTPUT LOW ADDRESS
                                <1>         IODELAY                             ; WAIT FOR I/O
2968 00004ADC EB00              <2>  jmp short $+2
2969 00004ADE EB00              <2>  jmp short $+2
00004AE0 88E0                   <1>         MOV     AL,AH
00004AE2 E604                   <1>         OUT     DMA+4,AL            ; OUTPUT HIGH ADDRESS
00004AE4 C1E810                 <1>         shr     eax, 16
                                <1>         IODELAY                             ; I/O WAIT STATE
2968 00004AE7 EB00              <2>  jmp short $+2
2969 00004AE9 EB00              <2>  jmp short $+2
00004AEB E681                   <1>         OUT     081H,AL                     ; OUTPUT highest BITS TO PAGE REGISTER
                                <1>         IODELAY
2968 00004AED EB00              <2>  jmp short $+2
2969 00004AEF EB00              <2>  jmp short $+2
00004AF1 6689C8                 <1>         mov     ax, cx              ; Byte count - 1
00004AF4 E605                   <1>         OUT     DMA+5,AL            ; LOW BYTE OF COUNT
                                <1>         IODELAY                             ; WAIT FOR I/O
2968 00004AF6 EB00              <2>  jmp short $+2
2969 00004AF8 EB00              <2>  jmp short $+2
00004AFA 88E0                   <1>         MOV     AL, AH
00004AFC E605                   <1>         OUT     DMA+5,AL            ; HIGH BYTE OF COUNT
                                <1>         IODELAY
2968 00004AFE EB00              <2>  jmp short $+2
2969 00004B00 EB00              <2>  jmp short $+2
00004B02 FB                     <1>         STI                         ; RE-ENABLE INTERRUPTS
00004B03 B002                   <1>         MOV     AL, 2               ; MODE FOR 8237
00004B05 E60A                   <1>         OUT     DMA+10,AL           ; INITIALIZE THE DISKETTE CHANNEL
                                <1>
00004B07 F8                     <1>         clc     ; 04/02/2016
00004B08 C3                     <1>         retn
                                <1>
                                <1> dma_bnd_err_stc:
00004B09 F9                     <1>         stc
                                <1> dma_bnd_err:
00004B0A C605[78890100]09       <1>         MOV     byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
00004B11 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
00004B12 8B5504         <1>       mov    edx, [ebp+4]       ; Buffer address
00004B15 F7C20000F0FF   <1>       test   edx, 0FFF00000h          ; 16 MB limit
00004B1B 75EC           <1>       jnz    short dma_bnd_err_stc
                        <1>       ;
                        <1>       ;push  dx                 ; *
                        <1>       ; 11/04/2021
00004B1D 52             <1>       push   edx
00004B1E B204           <1>       mov    DL, 4              ; SECTORS/TRACK VALUE IN PARM TABLE
00004B20 E8C7020000     <1>       call   GET_PARM           ; "
00004B25 88E0           <1>       mov    al, ah             ; AL = SECTORS/TRACK VALUE
00004B27 28E4           <1>       sub    ah, ah             ; AX = SECTORS/TRACK VALUE
00004B29 66C1E002       <1>       shl    ax, 2              ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
00004B2D 6648           <1>       dec    ax                 ; -1 FOR DMA VALUE
00004B2F 6689C1         <1>       mov    cx, ax
                        <1>       ;pop   dx                 ; *
                        <1>       ; 11/04/2021
00004B32 5A             <1>       pop    edx
00004B33 6601CA         <1>       add    dx, cx             ; check for overflow
00004B36 72D2           <1>       jc     short dma_bnd_err
                        <1>       ;
00004B38 6629CA         <1>       sub    dx, cx             ; Restore start address
                        <1>       ;
```

```
4807 00004B3B B04A              <1>       MOV    AL, 04AH            ; WILL WRITE TO THE DISKETTE
4808 00004B3D FA                <1>       CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
4809 00004B3E E60C              <1>       OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
4810                            <1>       IODELAY                          ; WAIT FOR I/O
2968 00004B40 EB00              <2>  jmp short $+2
2969 00004B42 EB00              <2>  jmp short $+2
4811 00004B44 E60B              <1>       OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
4812 00004B46 89D0              <1>       mov    eax, edx            ; Buffer address
4813 00004B48 E604              <1>       OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
4814                            <1>       IODELAY                          ; WAIT FOR I/O
2968 00004B4A EB00              <2>  jmp short $+2
2969 00004B4C EB00              <2>  jmp short $+2
4815 00004B4E 88E0              <1>       MOV    AL,AH
4816 00004B50 E604              <1>       OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
4817 00004B52 C1E810            <1>       shr    eax, 16
4818                            <1>       IODELAY                          ; I/O WAIT STATE
2968 00004B55 EB00              <2>  jmp short $+2
2969 00004B57 EB00              <2>  jmp short $+2
4819 00004B59 E681              <1>       OUT    081H,AL                   ; OUTPUT highest BITS TO PAGE REGISTER
4820                            <1>       IODELAY
2968 00004B5B EB00              <2>  jmp short $+2
2969 00004B5D EB00              <2>  jmp short $+2
4821 00004B5F 6689C8            <1>       mov    ax, cx              ; Byte count - 1
4822 00004B62 E605              <1>       OUT    DMA+5,AL            ; LOW BYTE OF COUNT
4823                            <1>       IODELAY                          ; WAIT FOR I/O
2968 00004B64 EB00              <2>  jmp short $+2
2969 00004B66 EB00              <2>  jmp short $+2
4824 00004B68 88E0              <1>       MOV    AL, AH
4825 00004B6A E605              <1>       OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
4826                            <1>       IODELAY
2968 00004B6C EB00              <2>  jmp short $+2
2969 00004B6E EB00              <2>  jmp short $+2
4827 00004B70 FB                <1>       STI                        ; RE-ENABLE INTERRUPTS
4828 00004B71 B002              <1>       MOV    AL, 2               ; MODE FOR 8237
4829 00004B73 E60A              <1>       OUT    DMA+10,AL           ; INITIALIZE THE DISKETTE CHANNEL
4830 00004B75 C3                <1>       retn
4831                            <1>
4832                            <1> ;; 08/02/2015 - Protected Mode Modification
4833                            <1> ;;    MOV    AL, 04AH            ; WILL WRITE TO THE DISKETTE
4834                            <1> ;;    CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
4835                            <1> ;;    OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
4836                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
4837                            <1> ;;    IODELAY
4838                            <1> ;;    OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
4839                            <1> ;;    ;MOV   AX,ES               ; GET THE ES VALUE
4840                            <1> ;;    ;ROL   AX,4                ; ROTATE LEFT
4841                            <1> ;;    ;MOV   CH,AL               ; GET HIGHEST NIBBLE OF ES TO CH
4842                            <1> ;;    ;AND   AL,11110000B        ; ZERO THE LOW NIBBLE FROM SEGMENT
4843                            <1> ;;    ;ADD   AX,[BP+2]           ; TEST FOR CARRY FROM ADDITION
4844                            <1> ;;    ;JNC   short J33A
4845                            <1> ;;    ;INC   CH                  ; CARRY MEANS HIGH 4 BITS MUST BE INC
4846                            <1> ;;    mov    eax, [ebp+4] ; 08/02/2015
4847                            <1> ;;;J33A:
4848                            <1> ;;    PUSH   eAX ; 08/02/2015    ; SAVE START ADDRESS
4849                            <1> ;;    OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
4850                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
4851                            <1> ;;    IODELAY
4852                            <1> ;;    MOV    AL,AH
4853                            <1> ;;    OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
4854                            <1> ;;    shr    eax, 16 ; 08/02/2015
4855                            <1> ;;    ;MOV   AL,CH               ; GET HIGH 4 BITS
4856                            <1> ;;    ;JMP   $+2                 ; I/O WAIT STATE
4857                            <1> ;;    IODELAY
4858                            <1> ;;    ;AND   AL,00001111B
4859                            <1> ;;    OUT    081H,AL                  ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
4860                            <1> ;;
4861                            <1> ;;;----- DETERMINE COUNT
4862                            <1> ;;    sub    eax, eax ; 08/02/2015
4863                            <1> ;;    MOV    DL, 4               ; SECTORS/TRACK VALUE IN PARM TABLE
4864                            <1> ;;    CALL   GET_PARM            ; "
4865                            <1> ;;    XCHG   AL, AH              ; AL = SECTORS/TRACK VALUE
4866                            <1> ;;    SUB    AH, AH              ; AX = SECTORS/TRACK VALUE
4867                            <1> ;;    SHL    AX, 2               ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
4868                            <1> ;;    DEC    AX                  ; -1 FOR DMA VALUE
4869                            <1> ;;    PUSH   eAX    ; 08/02/2015 ; SAVE # OF BYTES TO BE TRANSFERED
4870                            <1> ;;    OUT    DMA+5,AL            ; LOW BYTE OF COUNT
4871                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
4872                            <1> ;;    IODELAY
4873                            <1> ;;    MOV    AL, AH
4874                            <1> ;;    OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
4875                            <1> ;;    STI                        ; RE-ENABLE INTERRUPTS
4876                            <1> ;;    POP    eCX    ; 08/02/2015 ; RECOVER COUNT VALUE
4877                            <1> ;;    POP    eAX    ; 08/02/2015 ; RECOVER ADDRESS VALUE
4878                            <1> ;;    ;ADD   AX, CX              ; ADD, TEST FOR 64K OVERFLOW
4879                            <1> ;;    add    ecx, eax ; 08/02/2015
4880                            <1> ;;    MOV    AL, 2               ; MODE FOR 8237
4881                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
4882                            <1> ;;    SIODELAY
4883                            <1> ;;    OUT    DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
4884                            <1> ;;    ;JNC   short FMTDMA_OK           ; CHECK FOR ERROR
4885                            <1> ;;    jc     short fmtdma_bnd_err ; 08/02/2015
4886                            <1> ;;    and    ecx, 0FFF00000h  ; 16 MB limit
4887                            <1> ;;    jz     short FMTDMA_OK
4888                            <1> ;;    stc    ; 20/02/2015
4889                            <1> ;;fmtdma_bnd_err:
4890                            <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
4891                            <1> ;;FMTDMA_OK:
4892                            <1> ;;    RETn                       ; CY SET BY ABOVE IF ERROR
4893                            <1>
4894                            <1> ;--------------------------------------------------------------------------------
4895                            <1> ; NEC_INIT
4896                            <1> ;     THIS ROUTINE SEEKS TO THE REQUESTED TRACK AND INITIALIZES
4897                            <1> ;     THE NEC FOR THE READ/WRITE/VERIFY/FORMAT OPERATION.
4898                            <1> ;
4899                            <1> ; ON ENTRY:  AH = NEC COMMAND TO BE PERFORMED
```

```
                        <1> ;
                        <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1> NEC_INIT:
                        <1>       ;PUSH  AX                 ; SAVE NEC COMMAND
                        <1>       ; 11/04/2021
00004B76 50             <1>       push   eax
00004B77 E8B5020000     <1>       CALL   MOTOR_ON           ; TURN MOTOR ON FOR SPECIFIC DRIVE
                        <1>
                        <1> ;-----      DO THE SEEK OPERATION
                        <1>
00004B7C 8A6D01         <1>       MOV    CH,[eBP+1]         ; CH = TRACK #
00004B7F E8A8030000     <1>       CALL   SEEK               ; MOVE TO CORRECT TRACK
                        <1>       ;POP   AX                 ; RECOVER COMMAND
                        <1>       ; 11/04/2021
00004B84 58             <1>       pop    eax
00004B85 721E           <1>       JC     short ER_1         ; ERROR ON SEEK
00004B87 BB[A54B0000]   <1>       MOV    eBX, ER_1          ; LOAD ERROR ADDRESS
00004B8C 53             <1>       PUSH   eBX                ; PUSH NEC_OUT ERROR RETURN
                        <1>
                        <1> ;-----      SEND OUT THE PARAMETERS TO THE CONTROLLER
                        <1>
00004B8D E860030000     <1>       CALL   NEC_OUTPUT         ; OUTPUT THE OPERATION COMMAND
00004B92 6689F0         <1>       MOV    AX,SI              ; AH = HEAD #
00004B95 89FB           <1>       MOV    eBX,eDI                  ; BL = DRIVE #
00004B97 C0E402         <1>       SAL    AH,2               ; MOVE IT TO BIT 2
00004B9A 80E404         <1>       AND    AH,00000100B       ; ISOLATE THAT BIT
00004B9D 08DC           <1>       OR     AH,BL              ; OR IN THE DRIVE NUMBER
00004B9F E84E030000     <1>       CALL   NEC_OUTPUT         ; FALL THRU CY SET IF ERROR
00004BA4 5B             <1>       POP    eBX                ; THROW AWAY ERROR RETURN
                        <1> ER_1:
00004BA5 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; RWV_COM
                        <1> ;     THIS ROUTINE SENDS PARAMETERS TO THE NEC SPECIFIC TO THE
                        <1> ;     READ/WRITE/VERIFY OPERATIONS.
                        <1> ;
                        <1> ; ON ENTRY: CS:BX = ADDRESS OF MEDIA/DRIVE PARAMETER TABLE
                        <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1> RWV_COM:
00004BA6 B8[F14B0000]   <1>       MOV    eAX, ER_2          ; LOAD ERROR ADDRESS
00004BAB 50             <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
00004BAC 8A6501         <1>       MOV    AH,[eBP+1]         ; OUTPUT TRACK #
00004BAF E83E030000     <1>       CALL   NEC_OUTPUT
00004BB4 6689F0         <1>       MOV    AX,SI              ; OUTPUT HEAD #
00004BB7 E836030000     <1>       CALL   NEC_OUTPUT
00004BBC 8A6500         <1>         MOV     AH,[eBP]                ; OUTPUT SECTOR #
00004BBF E82E030000     <1>       CALL   NEC_OUTPUT
00004BC4 B203           <1>       MOV    DL,3               ; BYTES/SECTOR PARAMETER FROM BLOCK
00004BC6 E821020000     <1>       CALL   GET_PARM           ; ... TO THE NEC
00004BCB E822030000     <1>       CALL   NEC_OUTPUT         ; OUTPUT TO CONTROLLER
00004BD0 B204           <1>       MOV    DL,4               ; EOT PARAMETER FROM BLOCK
00004BD2 E815020000     <1>       CALL   GET_PARM           ; ... TO THE NEC
00004BD7 E816030000     <1>       CALL   NEC_OUTPUT         ; OUTPUT TO CONTROLLER
00004BDC 8A6305         <1>         MOV     AH, [eBX+MD.GAP]        ; GET GAP LENGTH
                        <1> _R15:
00004BDF E80E030000     <1>       CALL   NEC_OUTPUT
00004BE4 B206           <1>       MOV    DL,6               ; DTL PARAMETER PROM BLOCK
00004BE6 E801020000     <1>       CALL   GET_PARM           ;  TO THE NEC
00004BEB E802030000     <1>       CALL   NEC_OUTPUT         ; OUTPUT TO CONTROLLER
00004BF0 58             <1>       POP    eAX                ; THROW AWAY ERROR EXIT
                        <1> ER_2:
00004BF1 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; NEC_TERM
                        <1> ;     THIS ROUTINE WAITS FOR THE OPERATION THEN ACCEPTS THE STATUS
                        <1> ;     FROM THE NEC FOR THE READ/WRITE/VERIFY/FORWAT OPERATION.
                        <1> ;
                        <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1> NEC_TERM:
                        <1>
                        <1> ;-----      LET THE OPERATION HAPPEN
                        <1>
00004BF2 56             <1>       PUSH   eSI                ; SAVE HEAD #, # OF SECTORS
00004BF3 E805040000     <1>       CALL   WAIT_INT           ; WAIT FOR THE INTERRUPT
00004BF8 9C             <1>       PUSHF
00004BF9 E82F040000     <1>       CALL   RESULTS                  ; GET THE NEC STATUS
00004BFE 724B           <1>       JC     short SET_END_POP
00004C00 9D             <1>       POPF
00004C01 723E           <1>       JC     short SET_END      ; LOOK FOR ERROR
                        <1>
                        <1> ;-----      CHECK THE RESULTS RETURNED BY THE CONTROLLER
                        <1>
00004C03 FC             <1>       CLD                       ; SET THE CORRECT DIRECTION
00004C04 BE[79890100]   <1>       MOV    eSI, NEC_STATUS          ; POINT TO STATUS FIELD
00004C09 AC             <1>       lodsb                     ; GET ST0
00004C0A 24C0           <1>       AND    AL,11000000B       ; TEST FOR NORMAL TERMINATION
00004C0C 7433           <1>       JZ     short SET_END
00004C0E 3C40           <1>       CMP    AL,01000000B       ; TEST FOR ABNORMAL TERMINATION
00004C10 7527           <1>       JNZ    short J18          ; NOT ABNORMAL, BAD NEC
                        <1>
                        <1> ;-----      ABNORMAL TERMINATION, FIND OUT WHY
                        <1>
00004C12 AC             <1>       lodsb                     ; GET ST1
00004C13 D0E0           <1>       SAL    AL,1               ; TEST FOR EDT FOUND
00004C15 B404           <1>       MOV    AH,RECORD_NOT_FND
00004C17 7222           <1>       JC     short J19
00004C19 C0E002         <1>       SAL    AL,2
00004C1C B410           <1>       MOV    AH,BAD_CRC
00004C1E 721B           <1>       JC     short J19
00004C20 D0E0           <1>       SAL    AL,1               ; TEST FOR DMA OVERRUN
```

```
5005 00004C22 B408              <1>       MOV    AH,BAD_DMA
5006 00004C24 7215              <1>       JC     short J19
5007 00004C26 C0E002            <1>       SAL    AL,2                ; TEST FOR RECORD NOT FOUND
5008 00004C29 B404              <1>       MOV    AH,RECORD_NOT_FND
5009 00004C2B 720E              <1>       JC     short J19
5010 00004C2D D0E0              <1>       SAL    AL,1
5011 00004C2F B403              <1>       MOV    AH,WRITE_PROTECT    ; TEST FOR WRITE_PROTECT
5012 00004C31 7208              <1>       JC     short J19
5013 00004C33 D0E0              <1>       SAL    AL,1                ; TEST MISSING ADDRESS MARK
5014 00004C35 B402              <1>       MOV    AH,BAD_ADDR_MARK
5015 00004C37 7202              <1>       JC     short J19
5016                            <1>
5017                            <1> ;-----       NEC MUST HAVE FAILED
5018                            <1> J18:
5019 00004C39 B420              <1>       MOV    AH,BAD_NEC
5020                            <1> J19:
5021 00004C3B 0825[78890100]    <1>       OR     [DSKETTE_STATUS], AH
5022                            <1> SET_END:
5023 00004C41 803D[78890100]01  <1>       CMP    byte [DSKETTE_STATUS], 1 ; SET ERROR CONDITION
5024 00004C48 F5                <1>       CMC
5025 00004C49 5E                <1>       POP    eSI
5026 00004C4A C3                <1>       RETn                       ; RESTORE HEAD #, # OF SECTORS
5027                            <1>
5028                            <1> SET_END_POP:
5029 00004C4B 9D                <1>       POPF
5030 00004C4C EBF3              <1>       JMP    SHORT SET_END
5031                            <1>
5032                            <1> ;--------------------------------------------------------------------------
5033                            <1> ; DSTATE:   ESTABLISH STATE UPON SUCCESSFUL OPERATION.
5034                            <1> ;--------------------------------------------------------------------------
5035                            <1> DSTATE:
5036 00004C4E 803D[78890100]00  <1>       CMP    byte [DSKETTE_STATUS],0    ; CHECK FOR ERROR
5037 00004C55 753E              <1>       JNZ    short SETBAC           ; IF ERROR JUMP
5038 00004C57 808F[85890100]10  <1>       OR     byte [DSK_STATE+eDI],MED_DET ; NO ERROR, MARK MEDIA AS DETERMINED
5039 00004C5E F687[85890100]04  <1>       TEST   byte [DSK_STATE+eDI],DRV_DET ; DRIVE DETERMINED ?
5040 00004C65 752E              <1>       JNZ    short SETBAC        ; IF DETERMINED NO TRY TO DETERMINE
5041 00004C67 8A87[85890100]    <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
5042 00004C6D 24C0              <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
5043 00004C6F 3C80              <1>       CMP    AL,RATE_250         ; RATE 250 ?
5044 00004C71 751B              <1>       JNE    short M_12          ; NO, MUST BE 1.2M OR 1.44M DRIVE
5045                            <1>
5046                            <1> ;-----       CHECK IF IT IS 1.44M
5047                            <1>
5048 00004C73 E86B010000        <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
5049                            <1>       ;;20/02/2015
5050                            <1>       ;;JC   short M_12          ; CMOS BAD
5051 00004C78 7414              <1>       jz     short M_12 ;; 20/02/2015
5052 00004C7A 3C04              <1>       CMP    AL, 4               ; 1.44MB DRIVE ?
5053 00004C7C 7410              <1>       JE     short M_12          ; YES
5054                            <1> M_720:
5055 00004C7E 80A7[85890100]FD  <1>       AND    byte [DSK_STATE+eDI], ~FMT_CAPA ; TURN OFF FORMAT CAPABILITY
5056 00004C85 808F[85890100]04  <1>       OR     byte [DSK_STATE+eDI],DRV_DET  ; MARK DRIVE DETERMINED
5057 00004C8C EB07              <1>       JMP    SHORT SETBAC        ; BACK
5058                            <1> M_12:
5059 00004C8E 808F[85890100]06  <1>       OR     byte [DSK_STATE+eDI],DRV_DET+FMT_CAPA
5060                            <1>                                  ; TURN ON DETERMINED & FMT CAPA
5061                            <1> SETBAC:
5062 00004C95 C3                <1>       RETn
5063                            <1>
5064                            <1> ;--------------------------------------------------------------------------
5065                            <1> ; RETRY
5066                            <1> ;     DETERMINES WHETHER A RETRY IS NECESSARY.
5067                            <1> ;     IF RETRY IS REQUIRED THEN STATE INFORMATION IS UPDATED FOR RETRY.
5068                            <1> ;
5069                            <1> ; ON EXIT:  CY = 1 FOR RETRY, CY = 0 FOR NO RETRY
5070                            <1> ;--------------------------------------------------------------------------
5071                            <1> RETRY:
5072 00004C96 803D[78890100]00  <1>       CMP    byte [DSKETTE_STATUS],0    ; GET STATUS OF OPERATION
5073 00004C9D 7445              <1>       JZ     short NO_RETRY             ; SUCCESSFUL OPERATION
5074 00004C9F 803D[78890100]80  <1>       CMP    byte [DSKETTE_STATUS],TIME_OUT ; IF TIME OUT NO RETRY
5075 00004CA6 743C              <1>       JZ     short NO_RETRY
5076 00004CA8 8AA7[85890100]    <1>       MOV    AH,[DSK_STATE+eDI]  ; GET MEDIA STATE OF DRIVE
5077 00004CAE F6C410            <1>       TEST   AH,MED_DET          ; ESTABLISHED/DETERMINED ?
5078 00004CB1 7531              <1>       JNZ    short NO_RETRY             ; IF ESTABLISHED STATE THEN TRUE ERROR
5079 00004CB3 80E4C0            <1>       AND    AH,RATE_MSK         ; ISOLATE RATE
5080 00004CB6 8A2D[80890100]    <1>       MOV    CH,[LASTRATE]       ; GET START OPERATION STATE
5081 00004CBC C0C504            <1>       ROL    CH,4                ; TO CORRESPONDING BITS
5082 00004CBF 80E5C0            <1>       AND    CH,RATE_MSK         ; ISOLATE RATE BITS
5083 00004CC2 38E5              <1>       CMP    CH,AH               ; ALL RATES TRIED
5084 00004CC4 741E              <1>       JE     short NO_RETRY             ; IF YES, THEN TRUE ERROR
5085                            <1>
5086                            <1> ;     SETUP STATE INDICATOR FOR RETRY ATTEMPT TO NEXT RATE
5087                            <1> ;      00000000B (500) -> 10000000B    (250)
5088                            <1> ;      10000000B (250) -> 01000000B    (300)
5089                            <1> ;      01000000B (300) -> 00000000B    (500)
5090                            <1>
5091 00004CC6 80FC01            <1>       CMP    AH,RATE_500+1       ; SET CY FOR RATE 500
5092 00004CC9 D0DC              <1>       RCR    AH,1                ; TO NEXT STATE
5093 00004CCB 80E4C0            <1>       AND    AH,RATE_MSK         ; KEEP ONLY RATE BITS
5094 00004CCE 80A7[85890100]1F  <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP)
5095                            <1>                                  ; RATE, DBL STEP OFF
5096 00004CD5 08A7[85890100]    <1>       OR     [DSK_STATE+eDI],AH  ; TURN ON NEW RATE
5097 00004CDB C605[78890100]00  <1>       MOV    byte [DSKETTE_STATUS],0    ; RESET STATUS FOR RETRY
5098 00004CE2 F9                <1>       STC                        ; SET CARRY FOR RETRY
5099 00004CE3 C3                <1>       RETn                       ; RETRY RETURN
5100                            <1>
5101                            <1> NO_RETRY:
5102 00004CE4 F8                <1>       CLC                        ; CLEAR CARRY NO RETRY
5103 00004CE5 C3                <1>       RETn                       ; NO RETRY RETURN
5104                            <1>
5105                            <1> ;--------------------------------------------------------------------------
5106                            <1> ; NUM_TRANS
5107                            <1> ;     THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT WERE
5108                            <1> ;     ACTUALLY TRANSFERRED TO/FROM THE DISKETTE.
5109                            <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:
00004CE6 30C0           <1>      XOR    AL,AL               ; CLEAR FOR ERROR
00004CE8 803D[78890100]00 <1>    CMP    byte [DSKETTE_STATUS],0   ; CHECK FOR ERROR
00004CEF 752C           <1>      JNZ    NT_OUT              ; IF ERROR 0 TRANSFERRED
00004CF1 B204           <1>      MOV    DL,4                ; SECTORS/TRACK OFFSET TO DL
00004CF3 E8F4000000     <1>      CALL   GET_PARM            ; AH = SECTORS/TRACK
00004CF8 8A1D[7E890100] <1>      MOV    BL, [NEC_STATUS+5]  ; GET ENDING SECTOR
00004CFE 6689F1         <1>      MOV    CX,SI               ; CH = HEAD # STARTED
00004D01 3A2D[7D890100] <1>      CMP    CH, [NEC_STATUS+4]  ; GET HEAD ENDED UP ON
00004D07 750D           <1>      JNZ    DIF_HD              ; IF ON SAME HEAD, THEN NO ADJUST
00004D09 8A2D[7C890100] <1>      MOV    CH, [NEC_STATUS+3]  ; GET TRACK ENDED UP ON
00004D0F 3A6D01         <1>      CMP    CH,[eBP+1]          ; IS IT ASKED FOR TRACK
00004D12 7404           <1>      JZ     short SAME_TRK            ; IF SAME TRACK NO INCREASE
00004D14 00E3           <1>      ADD    BL,AH               ; ADD SECTORS/TRACK
                        <1> DIF_HD:
00004D16 00E3           <1>      ADD    BL,AH               ; ADD SECTORS/TRACK
                        <1> SAME_TRK:
00004D18 2A5D00         <1>      SUB    BL,[eBP]            ; SUBTRACT START FROM END
00004D1B 88D8           <1>      MOV    AL,BL               ; TO AL
                        <1> NT_OUT:
00004D1D C3             <1>      RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; SETUP_END
                        <1> ;     RESTORES @MOTOR_COUNT TO PARAMETER PROVIDED IN TABLE
                        <1> ;     AND LOADS @DSKETTE_STATUS TO AH, AND SETS CY.
                        <1> ;
                        <1> ; ON EXIT:
                        <1> ;     AH, @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------
                        <1> SETUP_END:
00004D1E B202           <1>      MOV    DL,2                ; GET THE MOTOR WAIT PARAMETER
                        <1>      ;PUSH  AX                  ; SAVE NUMBER TRANSFERRED
                        <1>      ; 11/04/2021
00004D20 50             <1>      push   eax
00004D21 E8C6000000     <1>      CALL   GET_PARM
00004D26 8825[77890100] <1>      MOV    [MOTOR_COUNT],AH    ; STORE UPON RETURN
                        <1>      ;POP   AX                  ; RESTORE NUMBER TRANSFERRED
                        <1>      ; 11/04/2021
00004D2C 58             <1>      pop    eax
00004D2D 8A25[78890100] <1>      MOV    AH, [DSKETTE_STATUS]      ; GET STATUS OF OPERATION
00004D33 08E4           <1>      OR     AH,AH               ; CHECK FOR ERROR
00004D35 7402           <1>      JZ     short NUN_ERR       ; NO ERROR
00004D37 30C0           <1>      XOR    AL,AL               ; CLEAR NUMBER RETURNED
                        <1> NUN_ERR:
00004D39 80FC01         <1>      CMP    AH,1                ; SET THE CARRY FLAG TO INDICATE
00004D3C F5             <1>      CMC                        ; SUCCESS OR FAILURE
00004D3D C3             <1>      RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; SETUP_DBL
                        <1> ;     CHECK DOUBLE STEP.
                        <1> ;
                        <1> ; ON ENTRY : DI = DRIVE
                        <1> ;
                        <1> ; ON EXIT :  CY = 1 MEANS ERROR
                        <1> ;--------------------------------------------------------------------------
                        <1> SETUP_DBL:
00004D3E 8AA7[85890100] <1>      MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
00004D44 F6C410         <1>      TEST   AH,MED_DET          ; ESTABLISHED STATE ?
00004D47 757A           <1>      JNZ    short NO_DBL              ; IF ESTABLISHED THEN DOUBLE DONE
                        <1>
                        <1> ;-----      CHECK FOR TRACK 0 TO SPEED UP ACKNOWLEDGE OF UNFORMATTED DISKETTE
                        <1>
00004D49 C605[75890100]00 <1>    MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
00004D50 E8DC000000     <1>      CALL   MOTOR_ON            ; ENSURE MOTOR STAY ON
00004D55 B500           <1>      MOV    CH,0                ; LOAD TRACK 0
00004D57 E8D0010000     <1>      CALL   SEEK                ; SEEK TO TRACK 0
00004D5C E864000000     <1>      CALL   READ_ID                   ; READ ID FUNCTION
00004D61 7245           <1>      JC     short SD_ERR        ; IF ERROR NO TRACK 0
                        <1>
                        <1> ;-----      INITIALIZE START AND MAX TRACKS (TIMES 2 FOR BOTH HEADS)
                        <1>
00004D63 66B95004       <1>      MOV    CX,0450H            ; START, MAX TRACKS
00004D67 F687[85890100]01 <1>    TEST   byte [DSK_STATE+eDI],TRK_CAPA ; TEST FOR 80 TRACK CAPABILITY
00004D6E 7402           <1>      JZ     short CNT_OK        ; IF NOT COUNT IS SETUP
00004D70 B1A0           <1>      MOV    CL,0A0H                  ; MAXIMUM TRACK 1.2 MB
                        <1>
                        <1> ;     ATTEMPT READ ID OF ALL TRACKS, ALL HEADS UNTIL SUCCESS; UPON SUCCESS,
                        <1> ;     MUST SEE IF ASKED FOR TRACK IN SINGLE STEP MODE = TRACK ID READ; IF NOT
                        <1> ;     THEN SET DOUBLE STEP ON.
                        <1>
                        <1> CNT_OK:
                        <1>      ; 11/04/2021 (32 bit push/pop)
00004D72 C605[77890100]FF <1>      MOV     byte [MOTOR_COUNT], 0FFH ; ENSURE MOTOR STAYS ON FOR OPERATION
00004D79 51             <1>      PUSH   eCX                 ; SAVE TRACK, COUNT
00004D7A C605[78890100]00 <1>    MOV    byte [DSKETTE_STATUS],0   ; CLEAR STATUS, EXPECT ERRORS
00004D81 6631C0         <1>      XOR    AX,AX               ; CLEAR AX
00004D84 D0ED           <1>      SHR    CH,1                ; HALVE TRACK, CY = HEAD
00004D86 C0D003         <1>      RCL    AL,3                ; AX = HEAD IN CORRECT BIT
00004D89 50             <1>      PUSH   eAX                 ; SAVE HEAD
00004D8A E89D010000     <1>      CALL   SEEK                ; SEEK TO TRACK
00004D8F 58             <1>      POP    eAX                 ; RESTORE HEAD
00004D90 6609C7         <1>      OR     DI,AX               ; DI = HEAD OR'ED DRIVE
00004D93 E82D000000     <1>      CALL   READ_ID                   ; READ ID HEAD 0
00004D98 9C             <1>      PUSHF                      ; SAVE RETURN FROM READ_ID
00004D99 6681E7FB00     <1>      AND    DI,11111011B        ; TURN OFF HEAD 1 BIT
00004D9E 9D             <1>      POPF                       ; RESTORE ERROR RETURN
00004D9F 59             <1>      POP    eCX                 ; RESTORE COUNT
```

```
5215 00004DA0 7308               <1>       JNC    short DO_CHK        ; IF OK, ASKED = RETURNED TRACK ?
5216 00004DA2 FEC5               <1>       INC    CH                  ; INC FOR NEXT TRACK
5217 00004DA4 38CD               <1>       CMP    CH,CL               ; REACHED MAXIMUM YET
5218 00004DA6 75CA               <1>       JNZ    short CNT_OK        ; CONTINUE TILL ALL TRIED
5219                             <1>
5220                             <1> ;-----      FALL THRU, READ ID FAILED FOR ALL TRACKS
5221                             <1>
5222                             <1> SD_ERR:
5223 00004DA8 F9                 <1>       STC                        ; SET CARRY FOR ERROR
5224 00004DA9 C3                 <1>       RETn                       ; SETUP_DBL ERROR EXIT
5225                             <1>
5226                             <1> DO_CHK:
5227 00004DAA 8A0D[7C890100]     <1>       MOV    CL, [NEC_STATUS+3]  ; LOAD RETURNED TRACK
5228 00004DB0 888F[89890100]     <1>       MOV    [DSK_TRK+eDI], CL   ; STORE TRACK NUMBER
5229 00004DB6 D0ED               <1>       SHR    CH,1                ; HALVE TRACK
5230 00004DB8 38CD               <1>       CMP    CH,CL               ; IS IT THE SAME AS ASKED FOR TRACK
5231 00004DBA 7407               <1>       JZ     short NO_DBL        ; IF SAME THEN NO DOUBLE STEP
5232 00004DBC 808F[85890100]20   <1>       OR     byte [DSK_STATE+eDI],DBL_STEP ; TURN ON DOUBLE STEP REQUIRED
5233                             <1> NO_DBL:
5234 00004DC3 F8                 <1>       CLC                        ; CLEAR ERROR FLAG
5235 00004DC4 C3                 <1>       RETn
5236                             <1>
5237                             <1> ;--------------------------------------------------------------------------
5238                             <1> ; READ_ID
5239                             <1> ;     READ ID FUNCTION.
5240                             <1> ;
5241                             <1> ; ON ENTRY:  DI : BIT 2 = HEAD; BITS 1,0 = DRIVE
5242                             <1> ;
5243                             <1> ; ON EXIT:   DI : BIT 2 IS RESET, BITS 1,0 = DRIVE
5244                             <1> ;            @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
5245                             <1> ;--------------------------------------------------------------------------
5246                             <1> READ_ID:
5247 00004DC5 B8[E24D0000]       <1>       MOV    eAX, ER_3           ; MOVE NEC OUTPUT ERROR ADDRESS
5248 00004DCA 50                 <1>       PUSH   eAX
5249 00004DCB B44A               <1>       MOV    AH,4AH              ; READ ID COMMAND
5250 00004DCD E820010000         <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
5251 00004DD2 6689F8             <1>       MOV    AX,DI               ; DRIVE # TO AH, HEAD 0
5252 00004DD5 88C4               <1>       MOV    AH,AL
5253 00004DD7 E816010000         <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
5254 00004DDC E811FEFFFF         <1>       CALL   NEC_TERM            ; WAIT FOR OPERATION, GET STATUS
5255 00004DE1 58                 <1>       POP    eAX                 ; THROW AWAY ERROR ADDRESS
5256                             <1> ER_3:
5257 00004DE2 C3                 <1>       RETn
5258                             <1>
5259                             <1> ;--------------------------------------------------------------------------
5260                             <1> ; CMOS_TYPE
5261                             <1> ;     RETURNS DISKETTE TYPE FROM CMOS
5262                             <1> ;
5263                             <1> ; ON ENTRY:  DI = DRIVE #
5264                             <1> ;
5265                             <1> ; ON EXIT:   AL = TYPE; CY REFLECTS STATUS
5266                             <1> ;--------------------------------------------------------------------------
5267                             <1>
5268                             <1> CMOS_TYPE: ; 11/12/2014
5269 00004DE3 8A87[CE6C0000]     <1> mov   al, [eDI+fd0_type]
5270 00004DE9 20C0               <1> and   al, al ; 18/12/2014
5271 00004DEB C3                 <1> retn
5272                             <1>
5273                             <1> ;CMOS_TYPE:
5274                             <1> ;     MOV    AL, CMOS_DIAG       ; CMOS DIAGNOSTIC STATUS BYTE ADDRESS
5275                             <1> ;     CALL   CMOS_READ           ; GET CMOS STATUS
5276                             <1> ;     TEST   AL,BAD_BAT+BAD_CKSUM      ; BATTERY GOOD AND CHECKSUM VALID
5277                             <1> ;     STC                        ; SET CY = 1 INDICATING ERROR FOR RETURN
5278                             <1> ;     JNZ    short BAD_CM        ; ERROR IF EITHER BIT ON
5279                             <1> ;     MOV    AL,CMOS_DISKETTE    ; ADDRESS OF DISKETTE BYTE IN CMOS
5280                             <1> ;     CALL   CMOS_READ           ; GET DISKETTE BYTE
5281                             <1> ;     OR     DI,DI               ; SEE WHICH DRIVE IN QUESTION
5282                             <1> ;     JNZ    short TB            ; IF DRIVE 1, DATA IN LOW NIBBLE
5283                             <1> ;     ROR    AL,4                ; EXCHANGE NIBBLES IF SECOND DRIVE
5284                             <1> ;TB:
5285                             <1> ;     AND    AL,0FH              ; KEEP ONLY DRIVE DATA, RESET CY, 0
5286                             <1> ;BAD_CM:
5287                             <1> ;     RETn                       ; CY, STATUS OF READ
5288                             <1>
5289                             <1> ;--------------------------------------------------------------------------
5290                             <1> ; GET_PARM
5291                             <1> ;     THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE
5292                             <1> ;     BLOCK POINTED TO BY THE DATA VARIABLE @DISK_POINTER. A BYTE FROM
5293                             <1> ;     THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING
5294                             <1> ;     THE PARAMETER IN DL.
5295                             <1> ;
5296                             <1> ; ON ENTRY:  DL = INDEX OF BYTE TO BE FETCHED
5297                             <1> ;
5298                             <1> ; ON EXIT:   AH = THAT BYTE FROM BLOCK
5299                             <1> ;            AL,DH DESTROYED
5300                             <1> ;--------------------------------------------------------------------------
5301                             <1> GET_PARM:
5302                             <1>       ;PUSH  DS
5303 00004DEC 56                 <1>       PUSH   eSI
5304                             <1>       ;SUB   AX,AX               ; DS = 0, BIOS DATA AREA
5305                             <1>       ;MOV   DS,AX
5306                             <1>       ;;mov  ax, cs
5307                             <1>       ;;mov  ds, ax
5308                             <1>       ; 08/02/2015 (protected mode modifications, bx -> ebx)
5309 00004DED 87D3               <1>       XCHG   eDX,eBX                   ; BL = INDEX
5310                             <1>       ;SUB   BH,BH               ; BX = INDEX
5311 00004DEF 81E3FF000000       <1>       and    ebx, 0FFh
5312                             <1>       ;LDS   SI, [DISK_POINTER]  ; POINT TO BLOCK
5313                             <1>       ;
5314                             <1>       ; 17/12/2014
5315 00004DF5 66A1[BD6C0000]     <1>       mov    ax, [cfd] ; current (AL) and previous fd (AH)
5316 00004DFB 38E0               <1>       cmp    al, ah
5317 00004DFD 7425               <1>       je     short gpndc
5318 00004DFF A2[BE6C0000]       <1>       mov    [pfd], al ; current drive -> previous drive
5319 00004E04 53                 <1>       push   ebx ; 08/02/2015
```

```
5320 00004E05 88C3              <1>       mov    bl, al
5321                            <1>       ; 11/12/2014
5322 00004E07 8A83[CE6C0000]    <1>       mov    al, [eBX+fd0_type]  ; Drive type (0,1,2,3,4)
5323                            <1>       ; 18/12/2014
5324 00004E0D 20C0              <1>       and    al, al
5325 00004E0F 7507              <1>       jnz    short gpdtc
5326 00004E11 BB[A76C0000]      <1>       mov    ebx, MD_TBL6        ; 1.44 MB param. tbl. (default)
5327 00004E16 EB05              <1>         jmp     short gpdpu
5328                            <1> gpdtc:
5329 00004E18 E835F9FFFF        <1>       call   DR_TYPE_CHECK
5330                            <1>       ; cf = 1 -> eBX points to 1.44MB fd parameter table (default)
5331                            <1> gpdpu:
5332 00004E1D 891D[446C0000]    <1>       mov    [DISK_POINTER], ebx
5333 00004E23 5B                <1>       pop    ebx
5334                            <1> gpndc:
5335 00004E24 8B35[446C0000]    <1>       mov    esi, [DISK_POINTER] ; 08/02/2015, si -> esi
5336 00004E2A 8A241E            <1>       MOV    AH, [eSI+eBX]       ; GET THE WORD
5337 00004E2D 87D3              <1>       XCHG   eDX,eBX                   ; RESTORE BX
5338 00004E2F 5E                <1>       POP    eSI
5339                            <1>       ;POP   DS
5340 00004E30 C3                <1>       RETn
5341                            <1>
5342                            <1> ;--------------------------------------------------------------------------
5343                            <1> ; MOTOR_ON
5344                            <1> ;     TURN MOTOR ON AND WAIT FOR MOTOR START UP TIME. THE @MOTOR_COUNT
5345                            <1> ;     IS REPLACED WITH A SUFFICIENTLY HIGH NUMBER (0FFH) TO ENSURE
5346                            <1> ;     THAT THE MOTOR DOES NOT GO OFF DURING THE OPERATION. IF THE
5347                            <1> ;     MOTOR NEEDED TO BE TURNED ON, THE MULTI-TASKING HOOK FUNCTION
5348                            <1> ;     (AX=90FDH, INT 15) IS CALLED TELLING THE OPERATING SYSTEM
5349                            <1> ;     THAT THE BIOS IS ABOUT TO WAIT FOR MOTOR START UP. IF THIS
5350                            <1> ;     FUNCTION RETURNS WITH CY = 1, IT MEANS THAT THE MINIMUM WAIT
5351                            <1> ;     HAS BEEN COMPLETED. AT THIS POINT A CHECK IS MADE TO ENSURE
5352                            <1> ;     THAT THE MOTOR WASN'T TURNED OFF BY THE TIMER. IF THE HOOK DID
5353                            <1> ;     NOT WAIT, THE WAIT FUNCTION (AH=086H) IS CALLED TO WAIT THE
5354                            <1> ;     PRESCRIBED AMOUNT OF TIME. IF THE CARRY FLAG IS SET ON RETURN,
5355                            <1> ;     IT MEANS THAT THE FUNCTION IS IN USE AND DID NOT PERFORM THE
5356                            <1> ;     WAIT. A TIMER 1 WAIT LOOP WILL THEN DO THE WAIT.
5357                            <1> ;
5358                            <1> ; ON ENTRY:  DI = DRIVE #
5359                            <1> ; ON EXIT:   AX,CX,DX DESTROYED
5360                            <1> ;--------------------------------------------------------------------------
5361                            <1> MOTOR_ON:
5362 00004E31 53                <1>       PUSH   eBX                 ; SAVE REG.
5363 00004E32 E82A000000        <1>       CALL   TURN_ON                   ; TURN ON MOTOR
5364 00004E37 7226              <1>       JC     short MOT_IS_ON           ; IF CY=1 NO WAIT
5365 00004E39 E8B7F9FFFF        <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
5366 00004E3E E881F9FFFF        <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH,
5367                            <1>       ;CALL  TURN_ON            ; CHECK AGAIN IF MOTOR ON
5368                            <1>       ;JC    MOT_IS_ON          ; IF NO WAIT MEANS IT IS ON
5369                            <1> M_WAIT:
5370 00004E43 B20A              <1>       MOV    DL,10              ; GET THE MOTOR WAIT PARAMETER
5371 00004E45 E8A2FFFFFF        <1>       CALL   GET_PARM
5372                            <1>       ;MOV   AL,AH              ; AL = MOTOR WAIT PARAMETER
5373                            <1>       ;XOR   AH,AH              ; AX = MOTOR WAIT PARAMETER
5374                            <1>       ;CMP   AL,8               ; SEE IF AT LEAST A SECOND IS SPECIFIED
5375 00004E4A 80FC08            <1>       cmp    ah, 8
5376                            <1>       ;JAE   short GP2          ; IF YES, CONTINUE
5377 00004E4D 7702              <1>       ja     short J13
5378                            <1>       ;MOV   AL,8               ; ONE SECOND WAIT FOR MOTOR START UP
5379 00004E4F B408              <1>       mov    ah, 8
5380                            <1>
5381                            <1> ;-----      AS CONTAINS NUMBER OF 1/8 SECONDS (125000 MICROSECONDS) TO WAIT
5382                            <1> GP2:
5383                            <1> ;-----      FOLLOWING LOOPS REQUIRED WHEN RTC WAIT FUNCTION IS ALREADY IN USE
5384                            <1> J13:                           ; WAIT FOR 1/8 SECOND PER (AL)
5385                            <1>       ;MOV   eCX,8286           ; COUNT FOR 1/8 SECOND AT 15.085737 US
5386                            <1>       ; 11/04/2021
5387 00004E51 B947100000        <1>       mov    ecx, 4167 ; count of 30 micro seconds * (1/8)
5388 00004E56 E86ED5FFFF        <1>       CALL   WAITF              ; GO TO FIXED WAIT ROUTINE
5389                            <1>       ;DEC   AL                 ; DECREMENT TIME VALUE
5390 00004E5B FECC              <1>       dec    ah
5391 00004E5D 75F2              <1>       JNZ    short J13          ; ARE WE DONE YET
5392                            <1> MOT_IS_ON:
5393 00004E5F 5B                <1>       POP    eBX                ; RESTORE REG.
5394 00004E60 C3                <1>       RETn
5395                            <1>
5396                            <1> ;--------------------------------------------------------------------------
5397                            <1> ; TURN_ON
5398                            <1> ;     TURN MOTOR ON AND RETURN WAIT STATE.
5399                            <1> ;
5400                            <1> ; ON ENTRY:  DI = DRIVE #
5401                            <1> ;
5402                            <1> ; ON EXIT:   CY = 0 MEANS WAIT REQUIRED
5403                            <1> ;            CY = 1 MEANS NO WAIT REQUIRED
5404                            <1> ;            AX,BX,CX,DX DESTROYED
5405                            <1> ;--------------------------------------------------------------------------
5406                            <1> TURN_ON:
5407 00004E61 89FB              <1>       MOV    eBX,eDI                   ; BX = DRIVE #
5408 00004E63 88D9              <1>       MOV    CL,BL              ; CL = DRIVE #
5409 00004E65 C0C304            <1>       ROL    BL,4               ; BL = DRIVE SELECT
5410 00004E68 FA                <1>       CLI                       ; NO INTERRUPTS WHILE DETERMINING STATUS
5411 00004E69 C605[77890100]FF  <1>       MOV    byte [MOTOR_COUNT],0FFH   ; ENSURE MOTOR STAYS ON FOR OPERATION
5412 00004E70 A0[76890100]      <1>       MOV    AL, [MOTOR_STATUS] ; GET DIGITAL OUTPUT REGISTER REFLECTION
5413 00004E75 2430              <1>       AND    AL,00110000B       ; KEEP ONLY DRIVE SELECT BITS
5414 00004E77 B401              <1>       MOV    AH,1               ; MASK FOR DETERMINING MOTOR BIT
5415 00004E79 D2E4              <1>       SHL    AH,CL              ; AH = MOTOR ON, A=00000001, B=00000010
5416                            <1>
5417                            <1> ;  AL = DRIVE SELECT FROM @MOTOR_STATUS
5418                            <1> ;  BL = DRIVE SELECT DESIRED
5419                            <1> ;  AH = MOTOR ON MASK DESIRED
5420                            <1>
5421 00004E7B 38D8              <1>       CMP    AL,BL              ; REQUESTED DRIVE ALREADY SELECTED ?
5422 00004E7D 7508              <1>       JNZ    short TURN_IT_ON   ; IF NOT SELECTED JUMP
5423 00004E7F 8425[76890100]    <1>       TEST   AH, [MOTOR_STATUS] ; TEST MOTOR ON BIT
5424 00004E85 7535              <1>       JNZ    short NO_MOT_WAIT  ; JUMP IF MOTOR ON AND SELECTED
```

```
                                <1>
                                <1> TURN_IT_ON:
00004E87 08DC                   <1>       OR     AH,BL                ; AH = DRIVE SELECT AND MOTOR ON
00004E89 8A3D[76890100]         <1>       MOV    BH,[MOTOR_STATUS]    ; SAVE COPY OF @MOTOR_STATUS BEFORE
00004E8F 80E70F                 <1>       AND    BH,00001111B         ; KEEP ONLY MOTOR BITS
00004E92 8025[76890100]CF       <1>       AND    byte [MOTOR_STATUS],11001111B ; CLEAR OUT DRIVE SELECT
00004E99 0825[76890100]         <1>       OR     [MOTOR_STATUS],AH    ; OR IN DRIVE SELECTED AND MOTOR ON
00004E9F A0[76890100]           <1>       MOV    AL,[MOTOR_STATUS]    ; GET DIGITAL OUTPUT REGISTER REFLECTION
00004EA4 88C3                   <1>       MOV    BL,AL                ; BL=@MOTOR_STATUS AFTER, BH=BEFORE
00004EA6 80E30F                 <1>       AND    BL,00001111B         ; KEEP ONLY MOTOR BITS
00004EA9 FB                     <1>       STI                         ; ENABLE INTERRUPTS AGAIN
00004EAA 243F                   <1>       AND    AL,00111111B         ; STRIP AWAY UNWANTED BITS
00004EAC C0C004                 <1>       ROL    AL,4                 ; PUT BITS IN DESIRED POSITIONS
00004EAF 0C0C                   <1>       OR     AL,00001100B         ; NO RESET, ENABLE DMA/INTERRUPT
00004EB1 66BAF203               <1>       MOV    DX,03F2H             ; SELECT DRIVE AND TURN ON MOTOR
00004EB5 EE                     <1>       OUT    DX,AL
00004EB6 38FB                   <1>       CMP    BL,BH                ; NEW MOTOR TURNED ON ?
                                <1>       ;JZ    short NO_MOT_WAIT    ; NO WAIT REQUIRED IF JUST SELECT
00004EB8 7403                   <1>       je     short no_mot_w1 ; 27/02/2015
00004EBA F8                     <1>       CLC                         ; (re)SET CARRY MEANING WAIT
00004EBB C3                     <1>       RETn
                                <1>
                                <1> NO_MOT_WAIT:
00004EBC FB                     <1>       sti
                                <1> no_mot_w1: ; 27/02/2015
00004EBD F9                     <1>       STC                         ; SET NO WAIT REQUIRED
                                <1>       ;STI                        ; INTERRUPTS BACK ON
00004EBE C3                     <1>       RETn
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; HD_WAIT
                                <1> ;     WAIT FOR HEAD SETTLE TIME.
                                <1> ;
                                <1> ; ON ENTRY:  DI = DRIVE #
                                <1> ;
                                <1> ; ON EXIT:   AX,BX,CX,DX DESTROYED
                                <1> ;--------------------------------------------------------------------------------
                                <1> HD_WAIT:
00004EBF B209                   <1>       MOV    DL,9                 ; GET HEAD SETTLE PARAMETER
00004EC1 E826FFFFFF             <1>       CALL   GET_PARM
00004EC6 08E4                   <1>       or     ah, ah ; 17/12/2014
00004EC8 7519                   <1>       jnz    short DO_WAT
00004ECA F605[76890100]80       <1>         TEST    byte [MOTOR_STATUS],10000000B ; SEE IF A WRITE OPERATION
                                <1>       ;JZ    short ISNT_WRITE    ; IF NOT, DO NOT ENFORCE ANY VALUES
                                <1>       ;OR    AH,AH                ; CHECK FOR ANY WAIT?
                                <1>       ;JNZ   short DO_WAT         ; IF THERE DO NOT ENFORCE
00004ED1 741E                   <1>       jz     short HW_DONE
00004ED3 B40F                   <1>       MOV    AH,HD12_SETTLE            ; LOAD 1.2M HEAD SETTLE MINIMUM
00004ED5 8A87[85890100]         <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
00004EDB 24C0                   <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
00004EDD 3C80                   <1>       CMP    AL,RATE_250         ; 1.2 M DRIVE ?
00004EDF 7502                   <1>       JNZ    short DO_WAT        ; DEFAULT HEAD SETTLE LOADED
                                <1> ;GP3:
00004EE1 B414                   <1>       MOV    AH,HD320_SETTLE           ; USE 320/360 HEAD SETTLE
                                <1> ;     JMP    SHORT DO_WAT
                                <1>
                                <1> ;ISNT_WRITE:
                                <1> ;     OR     AH,AH               ; CHECK FOR NO WAIT
                                <1> ;     JZ     short HW_DONE       ; IF NOT WRITE AND 0 ITS OK
                                <1>
                                <1> ;-----      AH CONTAINS NUMBER OF MILLISECONDS TO WAIT
                                <1> DO_WAT:
                                <1> ;     MOV    AL,AH               ; AL = # MILLISECONDS
                                <1> ;     ;XOR   AH,AH               ; AX = # MILLISECONDS
                                <1> J29:                              ;      1 MILLISECOND LOOP
                                <1>       ;;mov  cx, WAIT_FDU_HEAD_SETTLE ; 33 ; 1 ms in 30 micro units.
                                <1>       ;MOV   eCX,66              ; COUNT AT 15.085737 US PER COUNT
                                <1>       ; 11/04/2021
00004EE3 B921000000             <1>       mov    ecx, WAIT_FDU_HEAD_SETTLE ; 33
00004EE8 E8DCD4FFFF             <1>       CALL   WAITF               ; DELAY FOR 1 MILLISECOND
                                <1>       ;DEC   AL                  ; DECREMENT THE COUNT
00004EED FECC                   <1>       dec    ah
00004EEF 75F2                   <1>       JNZ    short J29           ; DO AL MILLISECOND # OF TIMES
                                <1> HW_DONE:
00004EF1 C3                     <1>       RETn
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; NEC_OUTPUT
                                <1> ;     THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
                                <1> ;     FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
                                <1> ;     TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE AMOUNT
                                <1> ;     OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
                                <1> ;
                                <1> ; ON ENTRY:  AH = BYTE TO BE OUTPUT
                                <1> ;
                                <1> ; ON EXIT:   CY = 0  SUCCESS
                                <1> ;            CY = 1  FAILURE -- DISKETTE STATUS UPDATED
                                <1> ;                    IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
                                <1> ;                    HIGHER THAN THE CALLER OF NEC OUTPUT. THIS REMOVES THE
                                <1> ;                    REQUIREMENT OF TESTING AFTER EVERY CALL OF NEC_OUTPUT.
                                <1> ;            AX,CX,DX DESTROYED
                                <1> ;--------------------------------------------------------------------------------
                                <1>
                                <1> ; 09/12/2014 [Erdogan Tan]
                                <1> ;     (from 'PS2 Hardware Interface Tech. Ref. May 88', Page 09-05.)
                                <1> ; Diskette Drive Controller Status Register (3F4h)
                                <1> ;     This read only register facilitates the transfer of data between
                                <1> ;     the system microprocessor and the controller.
                                <1> ; Bit 7 - When set to 1, the Data register is ready to transfer data
                                <1> ;       with the system micrprocessor.
                                <1> ; Bit 6 - The direction of data transfer. If this bit is set to 0,
                                <1> ;       the transfer is to the controller.
                                <1> ; Bit 5 - When this bit is set to 1, the controller is in the non-DMA mode.
                                <1> ; Bit 4 - When this bit is set to 1, a Read or Write command is being executed.
                                <1> ; Bit 3 - Reserved.
```

```
                        <1> ; Bit 2 - Reserved.
                        <1> ; Bit 1 - When this bit is set to 1, dskette drive 1 is in the seek mode.
                        <1> ; Bit 0 - When this bit is set to 1, dskette drive 1 is in the seek mode.
                        <1>
                        <1> ; Data Register (3F5h)
                        <1> ; This read/write register passes data, commands and parameters, and provides
                        <1> ; diskette status information.
                        <1>
                        <1> NEC_OUTPUT:
                        <1>       ;PUSH  BX                 ; SAVE REG.
00004EF2 66BAF403       <1>       MOV    DX,03F4H           ; STATUS PORT
                        <1>       ;MOV   BL,2               ; HIGH ORDER COUNTER
                        <1>       ;XOR   CX,CX              ; COUNT FOR TIME OUT
                        <1>       ; 16/12/2014
                        <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
00004EF6 B91B410000     <1>       mov    ecx, WAIT_FDU_SEND_LH   ; 16667 (27/02/2015)
                        <1> ;
                        <1> ;WFPS_OUTER_LP:
                        <1> ;     ;
                        <1> ;WFPS_CHECK_PORT:
                        <1> J23:
00004EFB EC             <1>       IN     AL,DX              ; GET STATUS
00004EFC 24C0           <1>       AND    AL,11000000B       ; KEEP STATUS AND DIRECTION
00004EFE 3C80           <1>       CMP    AL,10000000B       ; STATUS 1 AND DIRECTION 0 ?
00004F00 7418           <1>       JZ     short J27          ; STATUS AND DIRECTION OK
                        <1> WFPS_HI:
00004F02 E461           <1>       IN     AL, PORT_B   ;061h ; SYS1 ; wait for hi to lo
00004F04 A810           <1>       TEST   AL,010H                   ; transition on memory
00004F06 75FA           <1>       JNZ    SHORT WFPS_HI      ; refresh.
                        <1> WFPS_LO:
00004F08 E461           <1>       IN     AL, PORT_B         ; SYS1
00004F0A A810           <1>       TEST   AL,010H
00004F0C 74FA           <1>       JZ     SHORT WFPS_LO
                        <1>       ;LOOP  SHORT WFPS_CHECK_PORT
00004F0E 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>
00004F10 800D[78890100]80 <1>       OR     byte [DSKETTE_STATUS],TIME_OUT
                        <1>       ;POP   BX                 ; RESTORE REG.
00004F17 58             <1>       POP    eAX ; 08/02/2015   ; DISCARD THE RETURN ADDRESS
00004F18 F9             <1>       STC                       ; INDICATE ERROR TO CALLER
00004F19 C3             <1>       RETn
                        <1>
                        <1> ;-----      DIRECTION AND STATUS OK; OUTPUT BYTE
                        <1>
                        <1> J27:
00004F1A 88E0           <1>       MOV    AL,AH              ; GET BYTE TO OUTPUT
00004F1C 6642           <1>       INC    DX                 ; DATA PORT = STATUS PORT + 1
00004F1E EE             <1>       OUT    DX,AL              ; OUTPUT THE BYTE
                        <1>       ;;NEWIODELAY  ;; 27/02/2015
                        <1>       ; 27/02/2015
00004F1F 9C             <1>       PUSHF                     ; SAVE FLAGS
                        <1>       ;MOV   eCX, 3             ; 30 TO 45 MICROSECONDS WAIT FOR
                        <1>       ; 11/04/2021
00004F20 B902000000     <1>       mov    ecx, 2
00004F25 E89FD4FFFF     <1>       CALL   WAITF              ; NEC FLAGS UPDATE CYCLE
00004F2A 9D             <1>       POPF                      ; RESTORE FLAGS FOR EXIT
                        <1>       ;POP   BX                 ; RESTORE REG
00004F2B 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:
00004F2C 89FB           <1>       MOV    eBX,eDI                    ; BX = DRIVE #
00004F2E B001           <1>       MOV    AL,1                ; ESTABLISH MASK FOR RECALIBRATE TEST
00004F30 86CB           <1>       XCHG   CL,BL               ; SET DRIVE VALULE INTO CL
00004F32 D2C0           <1>       ROL    AL,CL               ; SHIFT MASK BY THE DRIVE VALUE
00004F34 86CB           <1>       XCHG   CL,BL               ; RECOVER TRACK VALUE
00004F36 8405[75890100] <1>       TEST   AL,[SEEK_STATUS]    ; TEST FOR RECALIBRATE REQUIRED
00004F3C 7526           <1>       JNZ    short J28A          ; JUMP IF RECALIBRATE NOT REQUIRED
                        <1>
00004F3E 0805[75890100] <1>       OR     [SEEK_STATUS],AL    ; TURN ON THE NO RECALIBRATE BIT IN FLAG
00004F44 E862000000     <1>       CALL   RECAL               ; RECALIBRATE DRIVE
00004F49 730E           <1>       JNC    short AFT_RECAL           ; RECALIBRATE DONE
                        <1>
                        <1> ;-----       ISSUE RECALIBRATE FOR 80 TRACK DISKETTES
                        <1>
00004F4B C605[78890100]00 <1>     MOV    byte [DSKETTE_STATUS],0    ; CLEAR OUT INVALID STATUS
00004F52 E854000000     <1>       CALL   RECAL               ; RECALIBRATE DRIVE
00004F57 7251           <1>       JC     short RB            ; IF RECALIBRATE FAILS TWICE THEN ERROR
                        <1>
                        <1> AFT_RECAL:
00004F59 C687[89890100]00 <1>       MOV     byte [DSK_TRK+eDI],0    ; SAVE NEW CYLINDER AS PRESENT POSITION
00004F60 08ED           <1>       OR     CH,CH               ; CHECK FOR SEEK TO TRACK 0
00004F62 743F           <1>       JZ     short DO_WAIT       ; HEAD SETTLE, CY = 0 IF JUMP
                        <1>
                        <1> ;-----       DRIVE IS IN SYNCHRONIZATION WITH CONTROLLER, SEEK TO TRACK
                        <1>
00004F64 F687[85890100]20 <1> J28A: TEST   byte [DSK_STATE+eDI],DBL_STEP ; CHECK FOR DOUBLE STEP REQUIRED
00004F6B 7402           <1>       JZ     short _R7           ; SINGLE STEP REQUIRED BYPASS DOUBLE
00004F6D D0E5           <1>       SHL    CH,1                ; DOUBLE NUMBER OF STEP TO TAKE
                        <1>
00004F6F 3AAF[89890100] <1> _R7:  CMP    CH, [DSK_TRK+eDI]   ; SEE IF ALREADY AT THE DESIRED TRACK
00004F75 7433           <1>       JE     short RB            ; IF YES, DO NOT NEED TO SEEK
                        <1>
00004F77 BA[AA4F0000]   <1>       MOV    eDX, NEC_ERR        ; LOAD RETURN ADDRESS
00004F7C 52             <1>       PUSH   eDX ; (*)           ; ON STACK FOR NEC OUTPUT ERROR
00004F7D 88AF[89890100] <1>       MOV    [DSK_TRK+eDI],CH    ; SAVE NEW CYLINDER AS PRESENT POSITION
00004F83 B40F           <1>       MOV    AH,0FH              ; SEEK COMMAND TO NEC
00004F85 E868FFFFFF     <1>       CALL   NEC_OUTPUT
00004F8A 89FB           <1>       MOV    eBX,eDI                 ; BX = DRIVE #
00004F8C 88DC           <1>       MOV    AH,BL               ; OUTPUT DRIVE NUMBER
00004F8E E85FFFFFFF     <1>       CALL   NEC_OUTPUT
00004F93 8AA7[89890100] <1>       MOV    AH, [DSK_TRK+eDI]   ; GET CYLINDER NUMBER
00004F99 E854FFFFFF     <1>       CALL   NEC_OUTPUT
00004F9E E827000000     <1>       CALL   CHK_STAT_2          ; ENDING INTERRUPT AND SENSE STATUS
                        <1>
                        <1> ;-----       WAIT FOR HEAD SETTLE
                        <1>
                        <1> DO_WAIT:
00004FA3 9C             <1>       PUSHF                      ; SAVE STATUS
00004FA4 E816FFFFFF     <1>       CALL   HD_WAIT                  ; WAIT FOR HEAD SETTLE TIME
00004FA9 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
00004FAA 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
00004FAB 51             <1>       push   ecx
00004FAC B8[C84F0000]   <1>       MOV    eAX, RC_BACK        ; LOAD NEC_OUTPUT ERROR
00004FB1 50             <1>       PUSH   eAX
00004FB2 B407           <1>       MOV    AH,07H              ; RECALIBRATE COMMAND
00004FB4 E839FFFFFF     <1>       CALL   NEC_OUTPUT
00004FB9 89FB           <1>       MOV    eBX,eDI                  ; BX = DRIVE #
00004FBB 88DC           <1>       MOV    AH,BL
00004FBD E830FFFFFF     <1>       CALL   NEC_OUTPUT          ; OUTPUT THE DRIVE NUMBER
00004FC2 E803000000     <1>       CALL   CHK_STAT_2          ; GET THE INTERRUPT AND SENSE INT STATUS
00004FC7 58             <1>       POP    eAX                 ; THROW AWAY ERROR
                        <1> RC_BACK:
                        <1>       ;POP   CX
                        <1>       ; 11/04/2021
00004FC8 59             <1>       pop    ecx
00004FC9 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:
00004FCA B8[F24F0000]   <1>       MOV     eAX, CS_BACK            ; LOAD NEC_OUTPUT ERROR ADDRESS
00004FCF 50             <1>       PUSH   eAX
00004FD0 E828000000     <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
00004FD5 721A           <1>       JC     short J34           ; IF ERROR, RETURN IT
00004FD7 B408           <1>       MOV    AH,08H              ; SENSE INTERRUPT STATUS COMMAND
```

```
5740 00004FD9 E814FFFFFF          <1>       CALL   NEC_OUTPUT
5741 00004FDE E84A000000          <1>       CALL   RESULTS                   ; READ IN THE RESULTS
5742 00004FE3 720C                <1>       JC     short J34
5743 00004FE5 A0[79890100]        <1>       MOV    AL,[NEC_STATUS]           ; GET THE FIRST STATUS BYTE
5744 00004FEA 2460                <1>       AND    AL,01100000B        ; ISOLATE THE BITS
5745 00004FEC 3C60                <1>       CMP    AL,01100000B        ; TEST FOR CORRECT VALUE
5746 00004FEE 7403                <1>       JZ     short J35           ; IF ERROR, GO MARK IT
5747 00004FF0 F8                  <1>       CLC                        ; GOOD RETURN
5748                              <1> J34:
5749 00004FF1 58                  <1>       POP    eAX                 ; THROW AWAY ERROR RETURN
5750                              <1> CS_BACK:
5751 00004FF2 C3                  <1>       RETn
5752                              <1> J35:
5753 00004FF3 800D[78890100]40    <1>       OR     byte [DSKETTE_STATUS], BAD_SEEK
5754 00004FFA F9                  <1>       STC                        ; ERROR RETURN CODE
5755 00004FFB EBF4                <1>       JMP    SHORT J34
5756                              <1>
5757                              <1> ;--------------------------------------------------------------------------------
5758                              <1> ; WAIT_INT
5759                              <1> ;     THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR A TIME OUT ROUTINE
5760                              <1> ;     TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE RETURNED
5761                              <1> ;     IF THE DRIVE IS NOT READY.
5762                              <1> ;
5763                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
5764                              <1> ;--------------------------------------------------------------------------------
5765                              <1>
5766                              <1> ; 17/12/2014
5767                              <1> ; 2.5 seconds waiting !
5768                              <1> ;(AWARD BIOS - 1999, WAIT_FDU_INT_LOW, WAIT_FDU_INT_HI)
5769                              <1> ; amount of time to wait for completion interrupt from NEC.
5770                              <1>
5771                              <1>
5772                              <1> WAIT_INT:
5773 00004FFD FB                  <1>       STI                        ; TURN ON INTERRUPTS, JUST IN CASE
5774 00004FFE F8                  <1>       CLC                        ; CLEAR TIMEOUT INDICATOR
5775                              <1>        ;MOV  BL,10               ; CLEAR THE COUNTERS
5776                              <1>        ;XOR  CX,CX               ; FOR 2 SECOND WAIT
5777                              <1>
5778                              <1>       ; Modification from AWARD BIOS - 1999 (ATORGS.ASM, WAIT
5779                              <1>       ;
5780                              <1>       ;WAIT_FOR_MEM:
5781                              <1>       ;     Waits for a bit at a specified memory location pointed
5782                              <1>       ;     to by ES:[DI] to become set.
5783                              <1>       ;INPUT:
5784                              <1>       ;     AH=Mask to test with.
5785                              <1>       ;     ES:[DI] = memory location to watch.
5786                              <1>       ;     BH:CX=Number of memory refresh periods to delay.
5787                              <1>       ;          (normally 30 microseconds per period.)
5788                              <1>
5789                              <1>       ; waiting for (max.) 2.5 secs in 30 micro units.
5790                              <1> ;     mov    cx, WAIT_FDU_INT_LO       ; 017798
5791                              <1> ;;    mov    bl, WAIT_FDU_INT_HI
5792                              <1> ;     mov    bl, WAIT_FDU_INT_HI + 1
5793                              <1>       ; 27/02/2015
5794 00004FFF B986450100          <1>       mov    ecx, WAIT_FDU_INT_LH      ; 83334 (2.5 seconds)
5795                              <1> WFMS_CHECK_MEM:
5796 00005004 F605[75890100]80    <1>       test   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
5797 0000500B 7516                <1>         jnz     short J37
5798                              <1> WFMS_HI:
5799 0000500D E461                <1>       IN     AL,PORT_B  ; 061h   ; SYS1, wait for lo to hi
5800 0000500F A810                <1>       TEST   AL,010H                   ; transition on memory
5801 00005011 75FA                <1>       JNZ    SHORT WFMS_HI       ; refresh.
5802                              <1> WFMS_LO:
5803 00005013 E461                <1>       IN     AL,PORT_B           ;SYS1
5804 00005015 A810                <1>       TEST   AL,010H
5805 00005017 74FA                <1>       JZ     SHORT WFMS_LO
5806 00005019 E2E9                <1>         LOOP    WFMS_CHECK_MEM
5807                              <1> ;WFMS_OUTER_LP:
5808                              <1> ;;    or     bl, bl              ; check outer counter
5809                              <1> ;;    jz     short J36A          ; WFMS_TIMEOUT
5810                              <1> ;     dec    bl
5811                              <1> ;     jz     short J36A
5812                              <1> ;     jmp    short WFMS_CHECK_MEM
5813                              <1>
5814                              <1>       ;17/12/2014
5815                              <1>       ;16/12/2014
5816                              <1> ;        mov     byte [wait_count], 0    ; Reset (INT 08H) counter
5817                              <1> ;J36:
5818                              <1> ;     TEST   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
5819                              <1> ;     JNZ    short J37
5820                              <1>       ;16/12/2014
5821                              <1>       ;LOOP  J36                 ; COUNT DOWN WHILE WAITING
5822                              <1>       ;DEC   BL                  ; SECOND LEVEL COUNTER
5823                              <1>       ;JNZ   short J36
5824                              <1> ;       cmp     byte [wait_count], 46   ; (46/18.2 seconds)
5825                              <1> ;     jb     short J36
5826                              <1>
5827                              <1> ;WFMS_TIMEOUT:
5828                              <1> ;J36A:
5829 0000501B 800D[78890100]80    <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; NOTHING HAPPENED
5830 00005022 F9                  <1>       STC                        ; ERROR RETURN
5831                              <1> J37:
5832 00005023 9C                  <1>       PUSHF                      ; SAVE CURRENT CARRY
5833 00005024 8025[75890100]7F    <1>       AND    byte [SEEK_STATUS], ~INT_FLAG ; TURN OFF INTERRUPT FLAG
5834 0000502B 9D                  <1>       POPF                       ; RECOVER CARRY
5835 0000502C C3                  <1>       RETn                       ; GOOD RETURN CODE
5836                              <1>
5837                              <1> ;--------------------------------------------------------------------------------
5838                              <1> ; RESULTS
5839                              <1> ;     THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER RETURNS
5840                              <1> ;     FOLLOWING AN INTERRUPT.
5841                              <1> ;
5842                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
5843                              <1> ;           AX,BX,CX,DX DESTROYED
5844                              <1> ;--------------------------------------------------------------------------------
```

```
5845                            <1> RESULTS:
5846 0000502D 57                <1>         PUSH   eDI
5847 0000502E BF[79890100]      <1>         MOV    eDI, NEC_STATUS          ; POINTER TO DATA AREA
5848 00005033 B307              <1>         MOV    BL,7                ; MAX STATUS BYTES
5849 00005035 66BAF403          <1>         MOV    DX,03F4H            ; STATUS PORT
5850                            <1>
5851                            <1> ;-----      WAIT FOR REQUEST FOR MASTER
5852                            <1>
5853                            <1> _R10:
5854                            <1>         ; 16/12/2014
5855                            <1>         ; wait for (max) 0.5 seconds
5856                            <1>         ;MOV   BH,2                ; HIGH ORDER COUNTER
5857                            <1>         ;XOR   CX,CX               ; COUNTER
5858                            <1>
5859                            <1>         ;Time to wait while waiting for each byte of NEC results = .5
5860                            <1>         ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
5861                            <1>         ; 27/02/2015
5862 00005039 B91B410000        <1>         mov    ecx, WAIT_FDU_RESULTS_LH ; 16667
5863                            <1>         ;mov   cx, WAIT_FDU_RESULTS_LO  ; 16667
5864                            <1>         ;mov   bh, WAIT_FDU_RESULTS_HI+1 ; 0+1
5865                            <1>
5866                            <1> WFPSR_OUTER_LP:
5867                            <1>         ;
5868                            <1> WFPSR_CHECK_PORT:
5869                            <1> J39:                               ; WAIT FOR MASTER
5870 0000503E EC                <1>         IN     AL,DX               ; GET STATUS
5871 0000503F 24C0              <1>         AND    AL,11000000B        ; KEEP ONLY STATUS AND DIRECTION
5872 00005041 3CC0              <1>         CMP    AL,11000000B        ; STATUS 1 AND DIRECTION 1 ?
5873 00005043 7418              <1>         JZ     short J42           ; STATUS AND DIRECTION OK
5874                            <1> WFPSR_HI:
5875 00005045 E461              <1>         IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
5876 00005047 A810              <1>         TEST   AL,010H                    ; transition on memory
5877 00005049 75FA              <1>         JNZ    SHORT WFPSR_HI             ; refresh.
5878                            <1> WFPSR_LO:
5879 0000504B E461              <1>         IN     AL, PORT_B          ; SYS1
5880 0000504D A810              <1>         TEST   AL,010H
5881 0000504F 74FA              <1>         JZ     SHORT WFPSR_LO
5882 00005051 E2EB              <1>           LOOP    WFPSR_CHECK_PORT
5883                            <1>         ;; 27/02/2015
5884                            <1>         ;;dec  bh
5885                            <1>         ;;jnz  short WFPSR_OUTER_LP
5886                            <1>         ;jmp   short WFPSR_TIMEOUT ; fail
5887                            <1>
5888                            <1>         ;;mov  byte [wait_count], 0
5889                            <1> ;J39:                              ; WAIT FOR MASTER
5890                            <1> ;       IN     AL,DX               ; GET STATUS
5891                            <1> ;       AND    AL,11000000B        ; KEEP ONLY STATUS AND DIRECTION
5892                            <1> ;       CMP    AL,11000000B        ; STATUS 1 AND DIRECTION 1 ?
5893                            <1> ;       JZ     short J42           ; STATUS AND DIRECTION OK
5894                            <1>         ;LOOP  J39                 ; LOOP TILL TIMEOUT
5895                            <1>         ;DEC   BH                  ; DECREMENT HIGH ORDER COUNTER
5896                            <1>         ;JNZ   short J39           ; REPEAT TILL DELAY DONE
5897                            <1>         ;
5898                            <1>         ;;cmp  byte [wait_count], 10  ; (10/18.2 seconds)
5899                            <1>         ;;jb   short J39
5900                            <1>
5901                            <1> ;WFPSR_TIMEOUT:
5902 00005053 800D[78890100]80  <1>         OR     byte [DSKETTE_STATUS],TIME_OUT
5903 0000505A F9                <1>         STC                        ; SET ERROR RETURN
5904 0000505B EB29              <1>         JMP    SHORT POPRES        ; POP REGISTERS AND RETURN
5905                            <1>
5906                            <1> ;-----      READ IN THE STATUS
5907                            <1>
5908                            <1> J42:
5909 0000505D EB00              <1>         JMP    $+2                 ; I/O DELAY
5910 0000505F 6642              <1>         INC    DX                  ; POINT AT DATA PORT
5911 00005061 EC                <1>         IN     AL,DX               ; GET THE DATA
5912                            <1>         ; 16/12/2014
5913                            <1>         NEWIODELAY
2973 00005062 E6EB              <2>  out 0ebh,al
5914 00005064 8807              <1>           MOV     [eDI],AL                ; STORE THE BYTE
5915 00005066 47                <1>         INC    eDI                 ; INCREMENT THE POINTER
5916                            <1>         ; 16/12/2014
5917                            <1> ;       push   cx
5918                            <1> ;       mov    cx, 30
5919                            <1> ;wdw2:
5920                            <1> ;       NEWIODELAY
5921                            <1> ;       loop   wdw2
5922                            <1> ;       pop    cx
5923                            <1>
5924                            <1>         ;MOV   eCX,3               ; MINIMUM 24 MICROSECONDS FOR NEC
5925                            <1>         ; 11/04/2021
5926 00005067 B902000000        <1>         mov    ecx, 2
5927 0000506C E858D3FFFF        <1>         CALL   WAITF               ; WAIT 30 TO 45 MICROSECONDS
5928 00005071 664A              <1>         DEC    DX                  ; POINT AT STATUS PORT
5929 00005073 EC                <1>         IN     AL,DX               ; GET STATUS
5930                            <1>         ; 16/12/2014
5931                            <1>         NEWIODELAY
2973 00005074 E6EB              <2>  out 0ebh,al
5932                            <1>         ;
5933 00005076 A810              <1>         TEST   AL,00010000B        ; TEST FOR NEC STILL BUSY
5934 00005078 740C              <1>         JZ     short POPRES        ; RESULTS DONE ?
5935                            <1>
5936 0000507A FECB              <1>         DEC    BL                  ; DECREMENT THE STATUS COUNTER
5937 0000507C 75BB              <1>           JNZ     short _R10              ; GO BACK FOR MORE
5938 0000507E 800D[78890100]20  <1>         OR     byte [DSKETTE_STATUS],BAD_NEC ; TOO MANY STATUS BYTES
5939 00005085 F9                <1>         STC                        ; SET ERROR FLAG
5940                            <1>
5941                            <1> ;-----      RESULT OPERATION IS DONE
5942                            <1> POPRES:
5943 00005086 5F                <1>         POP    eDI
5944 00005087 C3                <1>         RETn                       ; RETURN WITH CARRY SET
5945                            <1>
5946                            <1> ;--------------------------------------------------------------------------------
5947                            <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:
00005088 E8A4FDFFFF     <1>       CALL   MOTOR_ON           ; TURN ON THE MOTOR IF OFF
0000508D 66BAF703       <1>       MOV    DX,03F7H           ; ADDRESS DIGITAL INPUT REGISTER
00005091 EC             <1>       IN     AL,DX              ; INPUT DIGITAL INPUT REGISTER
00005092 A880           <1>       TEST   AL,DSK_CHG         ; CHECK FOR DISK CHANGE LINE ACTIVE
00005094 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:
00005095 E897FDFFFF     <1>       CALL   MOTOR_ON           ; TURN ON MOTOR IF NOT ALREADY ON
0000509A E80CFFFFFF     <1>       CALL   RECAL              ; RECALIBRATE DRIVE
0000509F 724F           <1>       JC     short DD_BAC       ; ASSUME NO DRIVE PRESENT
000050A1 B530           <1>       MOV    CH,TRK_SLAP        ; SEEK TO TRACK 48
000050A3 E884FEFFFF     <1>       CALL   SEEK
000050A8 7246           <1>       JC     short DD_BAC       ; ERROR NO DRIVE
000050AA B50B           <1>       MOV    CH,QUIET_SEEK+1          ; SEEK TO TRACK 10
                        <1> SK_GIN:
000050AC FECD           <1>       DEC    CH                 ; DECREMENT TO NEXT TRACK
                        <1>       ;PUSH  CX                 ; SAVE TRACK
                        <1>       ; 11/04/2021
000050AE 51             <1>       push   ecx
000050AF E878FEFFFF     <1>       CALL   SEEK
000050B4 723B           <1>       JC     short POP_BAC      ; POP AND RETURN
000050B6 B8[F1500000]   <1>       MOV    eAX, POP_BAC       ; LOAD NEC OUTPUT ERROR ADDRESS
000050BB 50             <1>       PUSH   eAX
000050BC B404           <1>       MOV    AH,SENSE_DRV_ST          ; SENSE DRIVE STATUS COMMAND BYTE
000050BE E82FFEFFFF     <1>       CALL   NEC_OUTPUT         ; OUTPUT TO NEC
000050C3 6689F8         <1>       MOV    AX,DI              ; AL = DRIVE
000050C6 88C4           <1>       MOV    AH,AL              ; AH = DRIVE
000050C8 E825FEFFFF     <1>       CALL   NEC_OUTPUT         ; OUTPUT TO NEC
000050CD E85BFFFFFF     <1>       CALL   RESULTS                 ; GO GET STATUS
000050D2 58             <1>       POP    eAX                ; THROW AWAY ERROR ADDRESS
                        <1>       ;POP   CX                 ; RESTORE TRACK
                        <1>       ; 11/04/2021
000050D3 59             <1>       pop    ecx
000050D4 F605[79890100]10 <1>     TEST   byte [NEC_STATUS], HOME   ; TRACK 0 ?
000050DB 74CF           <1>       JZ     short SK_GIN       ; GO TILL TRACK 0
000050DD 08ED           <1>       OR     CH,CH              ; IS HOME AT TRACK 0
000050DF 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>
000050E1 808F[85890100]94 <1>     OR     byte [DSK_STATE+eDI], DRV_DET+MED_DET+RATE_250
000050E8 C3             <1>       RETn                      ; ALL INFORMATION SET
                        <1> IS_80:
000050E9 808F[85890100]01 <1>     OR     byte [DSK_STATE+eDI], TRK_CAPA ; SETUP 80 TRACK CAPABILITY
                        <1> DD_BAC:
000050F0 C3             <1>       RETn
                        <1> POP_BAC:
                        <1>       ;POP   CX                 ; THROW AWAY
                        <1>       ; 11/04/2021
000050F1 59             <1>       pop    ecx
000050F2 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
000050F3 50             <1>       push   eax
000050F4 1E             <1>       push   ds
000050F5 66B81000       <1>       mov    ax, KDATA
000050F9 8ED8           <1>       mov    ds, ax
000050FB 800D[75890100]80 <1>       OR     byte [SEEK_STATUS], INT_FLAG ; TURN ON INTERRUPT OCCURRED
00005102 B020           <1>       MOV    AL,EOI                 ; END OF INTERRUPT MARKER
00005104 E620           <1>       OUT    INTA00,AL          ; INTERRUPT CONTROL PORT
00005106 1F             <1>       pop    ds
                        <1>       ;POP   AX                 ; RECOVER REGISTER
                        <1>       ; 11/04/2021
00005107 58             <1>       pop    eax
00005108 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:
```

```
6053                                <1>       ;PUSH  AX                 ; SAVE REGISTERS
6054                                <1>       ;PUSH  BX
6055                                <1>       ;PUSH  CX
6056 00005109 52                   <1>       PUSH   eDX
6057                                <1>       ;PUSH  DI
6058                                <1>       ;;PUSH DS
6059                                <1>       ; 14/12/2014
6060                                <1>       ;mov   word [DISK_POINTER], MD_TBL6
6061                                <1>       ;mov   [DISK_POINTER+2], cs
6062                                <1>       ;
6063                                <1>       ;OR    byte [RTC_WAIT_FLAG], 1    ; NO RTC WAIT, FORCE USE OF LOOP
6064 0000510A 31FF                 <1>       XOR    eDI,eDI                    ; INITIALIZE DRIVE POINTER
6065 0000510C 66C705[85890100]00-  <1>       MOV    WORD [DSK_STATE],0  ; INITIALIZE STATES
6065 00005114 00                   <1>
6066 00005115 8025[80890100]33     <1>       AND    byte [LASTRATE],~(STRT_MSK+SEND_MSK) ; CLEAR START & SEND
6067 0000511C 800D[80890100]C0     <1>       OR     byte [LASTRATE],SEND_MSK ; INITIALIZE SENT TO IMPOSSIBLE
6068 00005123 C605[75890100]00     <1>       MOV    byte [SEEK_STATUS],0       ; INDICATE RECALIBRATE NEEDED
6069 0000512A C605[77890100]00     <1>       MOV    byte [MOTOR_COUNT],0       ; INITIALIZE MOTOR COUNT
6070 00005131 C605[76890100]00     <1>       MOV    byte [MOTOR_STATUS],0      ; INITIALIZE DRIVES TO OFF STATE
6071 00005138 C605[78890100]00     <1>       MOV    byte [DSKETTE_STATUS],0    ; NO ERRORS
6072                                <1>       ;
6073                                <1>       ; 28/02/2015
6074                                <1>       ;mov   word [cfd], 100h
6075 0000513F E868F2FFFF           <1>       call   DSK_RESET
6076 00005144 5A                   <1>       pop    edx
6077 00005145 F8                   <1>       clc    ; 29/05/2016
6078 00005146 C3                   <1>       retn
6079                                <1>
6080                                <1> ;SUP0:
6081                                <1> ;     CALL   DRIVE_DET           ; DETERMINE DRIVE
6082                                <1> ;     CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
6083                                <1> ;     ; 02/01/2015
6084                                <1> ;     ;INC   DI                  ; POINT TO NEXT DRIVE
6085                                <1> ;     ;CMP   DI,MAX_DRV          ; SEE IF DONE
6086                                <1> ;     ;JNZ   short SUP0          ; REPEAT FOR EACH ORIVE
6087                                <1> ;       cmp     byte [fd1_type], 0
6088                                <1> ;     jna    short sup1
6089                                <1> ;     or     di, di
6090                                <1> ;     jnz    short sup1
6091                                <1> ;     inc    di
6092                                <1> ;       jmp     short SUP0
6093                                <1> ;sup1:
6094                                <1> ;     MOV    byte [SEEK_STATUS],0       ; FORCE RECALIBRATE
6095                                <1> ;     ;AND   byte [RTC_WAIT_FLAG],0FEH ; ALLOW FOR RTC WAIT
6096                                <1> ;     CALL   SETUP_END           ; VARIOUS CLEANUPS
6097                                <1> ;     ;;POP  DS                  ; RESTORE CALLERS REGISTERS
6098                                <1> ;     ;POP   DI
6099                                <1> ;     POP    eDX
6100                                <1> ;     ;POP   CX
6101                                <1> ;     ;POP   BX
6102                                <1> ;     ;POP   AX
6103                                <1> ;     RETn
6104                                <1>
6105                                <1> ;////////////////////////////////////////////////////
6106                                <1> ;; END OF DISKETTE I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
6107                                <1>
6108                                <1> ; 11/04/2021
6109                                <1> ;int13h: ; 21/02/2015
6110                                <1>       ;pushfd
6111                                <1>       ;push  cs
6112                                <1>       ;;call       DISK_IO
6113                                <1>       ;;retn
6114                                <1>
6115                                <1> ;;;;;; DISK I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; 21/02/2015 ;;;
6116                                <1> ;////////////////////////////////////////////////////////////////////
6117                                <1>
6118                                <1> ; DISK I/O - Erdogan Tan (Retro UNIX 386 v1 project)
6119                                <1> ; 18/02/2016
6120                                <1> ; 17/02/2016
6121                                <1> ; 23/02/2015
6122                                <1> ; 21/02/2015 (unix386.s)
6123                                <1> ; 22/12/2014 - 14/02/2015 (dsectrm2.s)
6124                                <1> ;
6125                                <1> ; Original Source Code:
6126                                <1> ; DISK ----- 09/25/85 FIXED DISK BIOS
6127                                <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
6128                                <1> ;
6129                                <1> ; Modifications: by reference of AWARD BIOS 1999 (D1A0622)
6130                                <1> ;             Source Code - ATORGS.ASM, AHDSK.ASM
6131                                <1> ;
6132                                <1>
6133                                <1>
6134                                <1> ;The wait for controller to be not busy is 10 seconds.
6135                                <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
6136                                <1> ;;WAIT_HDU_CTLR_BUSY_LO   equ    1615h
6137                                <1> ;;WAIT_HDU_CTLR_BUSY_HI   equ      05h
6138                                <1> WAIT_HDU_CTRL_BUSY_LH     equ    51615h  ;21/02/2015
6139                                <1>
6140                                <1> ;The wait for controller to issue completion interrupt is 10 seconds.
6141                                <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
6142                                <1> ;;WAIT_HDU_INT_LO  equ    1615h
6143                                <1> ;;WAIT_HDU_INT_HI  equ      05h
6144                                <1> WAIT_HDU_INT_LH           equ    51615h ; 21/02/2015
6145                                <1>
6146                                <1> ;The wait for Data request on read and write longs is
6147                                <1> ;2000 us. (?)
6148                                <1> ;;WAIT_HDU_DRQ_LO  equ    1000   ; 03E8h
6149                                <1> ;;WAIT_HDU_DRQ_HI  equ    0
6150                                <1> WAIT_HDU_DRQ_LH           equ    1000   ; 21/02/2015
6151                                <1>
6152                                <1> ; Port 61h (PORT_B)
6153                                <1> SYS1        equ    61h    ; PORT_B  (diskette.inc)
6154                                <1>
6155                                <1> ; 23/12/2014
6156                                <1> %define CMD_BLOCK       eBP-8  ; 21/02/2015
```

```
6157                                    <1>
6158                                    <1>
6159                                    <1> ;--- INT 13H -----------------------------------------------------------------------
6160                                    <1> ;                                                                                :
6161                                    <1> ; FIXED DISK I/O INTERFACE                                                       :
6162                                    <1> ;                                                                                :
6163                                    <1> ;     THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS THROUGH               :
6164                                    <1> ;     THE IBM FIXED DISK CONTROLLER.                                             :
6165                                    <1> ;                                                                                :
6166                                    <1> ;     THE  BIOS  ROUTINES  ARE  MEANT  TO  BE  ACCESSED  THROUGH               :
6167                                    <1> ;     SOFTWARE  INTERRUPTS  ONLY.    ANY  ADDRESSES  PRESENT     IN             :
6168                                    <1> ;     THESE  LISTINGS  ARE  INCLUDED    ONLY  FOR  COMPLETENESS,               :
6169                                    <1> ;     NOT  FOR  REFERENCE.  APPLICATIONS   WHICH  REFERENCE  ANY               :
6170                                    <1> ;     ABSOLUTE  ADDRESSES  WITHIN  THE  CODE  SEGMENTS  OF  BIOS               :
6171                                    <1> ;     VIOLATE  THE  STRUCTURE  AND  DESIGN  OF  BIOS.                          :
6172                                    <1> ;                                                                                :
6173                                    <1> ;--------------------------------------------------------------------------------:
6174                                    <1> ;                                                                                :
6175                                    <1> ; INPUT  (AH)= HEX COMMAND VALUE                                                 :
6176                                    <1> ;                                                                                :
6177                                    <1> ;     (AH)= 00H  RESET DISK (DL = 80H,81H) / DISKETTE                            :
6178                                    <1> ;     (AH)= 01H  READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)            :
6179                                    <1> ;                NOTE: DL < 80H - DISKETTE                                       :
6180                                    <1> ;                      DL > 80H - DISK                                           :
6181                                    <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY                            :
6182                                    <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY                           :
6183                                    <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS                                      :
6184                                    <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK                                        :
6185                                    <1> ;     (AH)= 06H  UNUSED                                                          :
6186                                    <1> ;     (AH)= 07H  UNUSED                                                          :
6187                                    <1> ;     (AH)= 08H  RETURN THE CURRENT DRIVE PARAMETERS                               :
6188                                    <1> ;     (AH)= 09H  INITIALIZE DRIVE PAIR CHARACTERISTICS                           :
6189                                    <1> ;                INTERRUPT 41 POINTS TO DATA BLOCK FOR DRIVE 0                   :
6190                                    <1> ;                INTERRUPT 46 POINTS TO DATA BLOCK FOR DRIVE 1                   :
6191                                    <1> ;     (AH)= 0AH  READ LONG                                                       :
6192                                    <1> ;     (AH)= 0BH  WRITE LONG  (READ & WRITE LONG ENCOMPASS 512 + 4 BYTES ECC)     :
6193                                    <1> ;     (AH)= 0CH  SEEK                                                            :
6194                                    <1> ;     (AH)= 0DH  ALTERNATE DISK RESET (SEE DL)                                   :
6195                                    <1> ;     (AH)= 0EH  UNUSED                                                          :
6196                                    <1> ;     (AH)= 0FH  UNUSED                                                          :
6197                                    <1> ;     (AH)= 10H  TEST DRIVE READY                                                :
6198                                    <1> ;     (AH)= 11H  RECALIBRATE                                                     :
6199                                    <1> ;     (AH)= 12H  UNUSED                                                          :
6200                                    <1> ;     (AH)= 13H  UNUSED                                                          :
6201                                    <1> ;     (AH)= 14H  CONTROLLER INTERNAL DIAGNOSTIC                                  :
6202                                    <1> ;     (AH)= 15H  READ DASD TYPE                                                  :
6203                                    <1> ;                                                                                :
6204                                    <1> ;--------------------------------------------------------------------------------
6205                                    <1> ;                                                                                :
6206                                    <1> ;     REGISTERS USED FOR FIXED DISK OPERATIONS                                   :
6207                                    <1> ;                                                                                :
6208                                    <1> ;           (DL)   -  DRIVE NUMBER     (80H-81H FOR DISK. VALUE CHECKED)      :
6209                                    <1> ;           (DH)   -  HEAD NUMBER          (0-15 ALLOWED, NOT VALUE CHECKED)  :
6210                                    <1> ;           (CH)   -  CYLINDER NUMBER  (0-1023, NOT VALUE CHECKED)(SEE CL):
6211                                    <1> ;           (CL)   -  SECTOR NUMBER    (1-17, NOT VALUE CHECKED)               :
6212                                    <1> ;                                                                                :
6213                                    <1> ;                     NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED            :
6214                                    <1> ;                          IN THE HIGH 2 BITS OF THE CL REGISTER                 :
6215                                    <1> ;                          (10 BITS TOTAL)                                       :
6216                                    <1> ;                                                                                :
6217                                    <1> ;           (AL)   -  NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,           :
6218                                    <1> ;                                       FOR READ/WRITE LONG 1-79H)               :
6219                                    <1> ;                                                                                :
6220                                    <1> ;           (ES:BX) -  ADDRESS OF BUFFER FOR READS AND WRITES,                   :
6221                                    <1> ;                     (NOT REQUIRED FOR VERIFY)                                  :
6222                                    <1> ;                                                                                :
6223                                    <1> ;           FORMAT (AH=5) ES:BX POINTS TO A 512 BYTE BUFFER. THE FIRST           :
6224                                    <1> ;                     2*(SECTORS/TRACK) BYTES CONTAIN F,N FOR EACH SECTOR.:
6225                                    <1> ;                     F = 00H FOR A GOOD SECTOR                                  :
6226                                    <1> ;                         80H FOR A BAD SECTOR                                   :
6227                                    <1> ;                     N = SECTOR NUMBER                                          :
6228                                    <1> ;                     FOR AN INTERLEAVE OF 2 AND 17 SECTORS/TRACK                  :
6229                                    <1> ;                     THE TABLE SHOULD BE:                                       :
6230                                    <1> ;                                                                                :
6231                                    <1> ;                DB     00H,01H,00H,0AH,00H,02H,00H,0BH,00H,03H,00H,0CH           :
6232                                    <1> ;                DB     00H,04H,00H,0DH,00H,05H,00H,0EH,00H,06H,00H,0FH           :
6233                                    <1> ;                DB     00H,07H,00H,10H,00H,08H,00H,11H,00H,09H                   :
6234                                    <1> ;                                                                                :
6235                                    <1> ;--------------------------------------------------------------------------------
6236                                    <1>
6237                                    <1> ;--------------------------------------------------------------------------------
6238                                    <1> ; OUTPUT                                                                         :
6239                                    <1> ;     AH = STATUS OF CURRENT OPERATION                                           :
6240                                    <1> ;          STATUS BITS ARE DEFINED IN THE EQUATES BELOW                          :
6241                                    <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)                               :
6242                                    <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)                              :
6243                                    <1> ;                                                                                :
6244                                    <1> ;     NOTE:  ERROR 11H  INDICATES THAT THE DATA READ HAD A RECOVERABLE           :
6245                                    <1> ;            ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM.  THE DATA            :
6246                                    <1> ;            IS PROBABLY GOOD,   HOWEVER THE BIOS ROUTINE INDICATES AN           :
6247                                    <1> ;            ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE            :
6248                                    <1> ;            FOR ITSELF.  THE  ERROR  MAY  NOT  RECUR  IF  THE DATA IS           :
6249                                    <1> ;            REWRITTEN.                                                          :
6250                                    <1> ;                                                                                :
6251                                    <1> ;     IF DRIVE PARAMETERS WERE REQUESTED (DL >= 80H),                            :
6252                                    <1> ;        INPUT:                                                                  :
6253                                    <1> ;          (DL) = DRIVE NUMBER                                                   :
6254                                    <1> ;          ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)                                       :
6255                                    <1> ;           EBX = Buffer address for fixed disk parameters table (32 bytes)      :
6256                                    <1> ;        OUTPUT:                                                                 :
6257                                    <1> ;          (DL) = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (1-2)    :
6258                                    <1> ;               (CONTROLLER CARD ZERO TALLY ONLY)                                :
6259                                    <1> ;          (DH) = MAXIMUM USEABLE VALUE FOR HEAD NUMBER                          :
6260                                    <1> ;          (CH) = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER                        :
```

```
6261                                 <1> ;          (CL) = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER               :
6262                                 <1> ;               AND CYLINDER NUMBER HIGH BITS                           :
6263                                 <1> ;                                                                     :
6264                                 <1> ;     IF READ DASD TYPE WAS REQUESTED,                                :
6265                                 <1> ;                                                                     :
6266                                 <1> ;     AH = 0 - NOT PRESENT                                            :
6267                                 <1> ;          1 - DISKETTE - NO CHANGE LINE AVAILABLE                    :
6268                                 <1> ;          2 - DISKETTE - CHANGE LINE AVAILABLE                       :
6269                                 <1> ;          3 - FIXED DISK                                             :
6270                                 <1> ;                                                                     :
6271                                 <1> ;     CX,DX = NUMBER OF 512 BYTE BLOCKS WHEN AH = 3                   :
6272                                 <1> ;                                                                     :
6273                                 <1> ;     REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN        :
6274                                 <1> ;     INFORMATION.                                                    :
6275                                 <1> ;                                                                     :
6276                                 <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE        :
6277                                 <1> ;           ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.           :
6278                                 <1> ;                                                                     :
6279                                 <1> ;---------------------------------------------------------------------------
6280                                 <1>
6281                                 <1> SENSE_FAIL   EQU    0FFH          ; NOT IMPLEMENTED
6282                                 <1> NO_ERR       EQU    0E0H          ; STATUS ERROR/ERROR REGISTER=0
6283                                 <1> WRITE_FAULT  EQU    0CCH          ; WRITE FAULT ON SELECTED DRIVE
6284                                 <1> UNDEF_ERR    EQU    0BBH          ; UNDEFINED ERROR OCCURRED
6285                                 <1> NOT_RDY      EQU    0AAH          ; DRIVE NOT READY
6286                                 <1> TIME_OUT     EQU    80H           ; ATTACHMENT FAILED TO RESPOND
6287                                 <1> BAD_SEEK     EQU    40H           ; SEEK OPERATION FAILED
6288                                 <1> BAD_CNTLR    EQU    20H           ; CONTROLLER HAS FAILED
6289                                 <1> DATA_CORRECTED     EQU    11H          ; ECC CORRECTED DATA ERROR
6290                                 <1> BAD_ECC      EQU    10H           ; BAD ECC ON DISK READ
6291                                 <1> BAD_TRACK    EQU    0BH           ; NOT IMPLEMENTED
6292                                 <1> BAD_SECTOR   EQU    0AH           ; BAD SECTOR FLAG DETECTED
6293                                 <1> ;DMA_BOUNDARY      EQU    09H          ; DATA EXTENDS TOO FAR
6294                                 <1> INIT_FAIL    EQU    07H           ; DRIVE PARAMETER ACTIVITY FAILED
6295                                 <1> BAD_RESET    EQU    05H           ; RESET FAILED
6296                                 <1> ;RECORD_NOT_FND    EQU    04H          ; REQUESTED SECTOR NOT FOUND
6297                                 <1> ;BAD_ADDR_MARK     EQU    02H          ; ADDRESS MARK NOT FOUND
6298                                 <1> ;BAD_CMD     EQU    01H           ; BAD COMMAND PASSED TO DISK I/O
6299                                 <1>
6300                                 <1> ;--------------------------------------------------------
6301                                 <1> ;                                                       :
6302                                 <1> ; FIXED DISK PARAMETER TABLE                            :
6303                                 <1> ;  -  THE TABLE IS COMPOSED OF A BLOCK DEFINED AS:      :
6304                                 <1> ;                                                       :
6305                                 <1> ;  +0 (1 WORD) - MAXIMUM NUMBER OF CYLINDERS            :
6306                                 <1> ;  +2 (1 BYTE) - MAXIMUM NUMBER OF HEADS                :
6307                                 <1> ;  +3 (1 WORD) - NOT USED/SEE PC-XT                     :
6308                                 <1> ;  +5 (1 WORD) - STARTING WRITE PRECOMPENSATION CYL     :
6309                                 <1> ;  +7 (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH          :
6310                                 <1> ;  +8 (1 BYTE) - CONTROL BYTE                           :
6311                                 <1> ;                BIT   7 DISABLE RETRIES -OR-           :
6312                                 <1> ;                BIT   6 DISABLE RETRIES                :
6313                                 <1> ;                BIT   3 MORE THAN 8 HEADS              :
6314                                 <1> ;  +9 (3 BYTES)- NOT USED/SEE PC-XT                     :
6315                                 <1> ; +12 (1 WORD) - LANDING ZONE                           :
6316                                 <1> ; +14 (1 BYTE) - NUMBER OF SECTORS/TRACK                :
6317                                 <1> ; +15 (1 BYTE) - RESERVED FOR FUTURE USE                :
6318                                 <1> ;                                                       :
6319                                 <1> ;      - TO DYNAMICALLY DEFINE A SET OF PARAMETERS      :
6320                                 <1> ;        BUILD A TABLE FOR UP TO 15 TYPES AND PLACE     :
6321                                 <1> ;        THE CORRESPONDING VECTOR INTO INTERRUPT 41     :
6322                                 <1> ;        FOR DRIVE 0 AND INTERRUPT 46 FOR DRIVE 1.      :
6323                                 <1> ;                                                       :
6324                                 <1> ;--------------------------------------------------------
6325                                 <1>
6326                                 <1> ;--------------------------------------------------------
6327                                 <1> ;                                                       :
6328                                 <1> ; HARDWARE SPECIFIC VALUES                              :
6329                                 <1> ;                                                       :
6330                                 <1> ;  -  CONTROLLER I/O PORT                               :
6331                                 <1> ;                                                       :
6332                                 <1> ;     > WHEN READ FROM:                                 :
6333                                 <1> ;     HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU)        :
6334                                 <1> ;     HF_PORT+1 - GET ERROR REGISTER                    :
6335                                 <1> ;     HF_PORT+2 - GET SECTOR COUNT                      :
6336                                 <1> ;     HF_PORT+3 - GET SECTOR NUMBER                     :
6337                                 <1> ;     HF_PORT+4 - GET CYLINDER LOW                      :
6338                                 <1> ;     HF_PORT+5 - GET CYLINDER HIGH (2 BITS)            :
6339                                 <1> ;     HF_PORT+6 - GET SIZE/DRIVE/HEAD                 :
6340                                 <1> ;     HF_PORT+7 - GET STATUS REGISTER                 :
6341                                 <1> ;                                                       :
6342                                 <1> ;     > WHEN WRITTEN TO:                                :
6343                                 <1> ;     HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER)       :
6344                                 <1> ;     HF_PORT+1 - SET PRECOMPENSATION CYLINDER          :
6345                                 <1> ;     HF_PORT+2 - SET SECTOR COUNT                      :
6346                                 <1> ;     HF_PORT+3 - SET SECTOR NUMBER                     :
6347                                 <1> ;     HF_PORT+4 - SET CYLINDER LOW                      :
6348                                 <1> ;     HF_PORT+5 - SET CYLINDER HIGH (2 BITS)            :
6349                                 <1> ;     HF_PORT+6 - SET SIZE/DRIVE/HEAD                 :
6350                                 <1> ;     HF_PORT+7 - SET COMMAND REGISTER                :
6351                                 <1> ;                                                       :
6352                                 <1> ;--------------------------------------------------------
6353                                 <1>
6354                                 <1> ;HF_PORT     EQU    01F0H  ; DISK PORT
6355                                 <1> ;HF1_PORT    equ    0170h
6356                                 <1> ;HF_REG_PORT EQU    03F6H
6357                                 <1> ;HF1_REG_PORT      equ    0376h
6358                                 <1>
6359                                 <1> HDC1_BASEPORT      equ    1F0h
6360                                 <1> HDC2_BASEPORT      equ    170h
6361                                 <1>
6362 00005147 90                    <1> align 2
6363                                 <1>
6364                                 <1> ;-----             STATUS REGISTER
6365                                 <1>
```

```
                        <1> ST_ERROR     EQU    00000001B    ;
                        <1> ST_INDEX     EQU    00000010B    ;
                        <1> ST_CORRCTD   EQU    00000100B    ; ECC CORRECTION SUCCESSFUL
                        <1> ST_DRQ              EQU    00001000B    ;
                        <1> ST_SEEK_COMPL       EQU    00010000B    ; SEEK COMPLETE
                        <1> ST_WRT_FLT   EQU    00100000B    ; WRITE FAULT
                        <1> ST_READY     EQU    01000000B    ;
                        <1> ST_BUSY      EQU    10000000B    ;
                        <1>
                        <1> ;-----              ERROR REGISTER
                        <1>
                        <1> ERR_DAM      EQU    00000001B    ; DATA ADDRESS MARK NOT FOUND
                        <1> ERR_TRK_0    EQU    00000010B    ; TRACK 0 NOT FOUND ON RECAL
                        <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
00005148 C705[90890100]0000-    <1>       mov    dword [HDPM_TBL_VEC], (DPT_SEGM*16)+HD0_DPT
00005150 0900                   <1>
                        <1>       ;mov   word [HDPS_TBL_VEC],HD1_DPT      ; PARM TABLE DRIVE 81h
                        <1>       ;mov   word [HDPS_TBL_VEC+2],DPT_SEGM
00005152 C705[94890100]2000-    <1>       mov    dword [HDPS_TBL_VEC], (DPT_SEGM*16)+HD1_DPT
```

```
6469 0000515A 0900                <1>
6470                              <1>       ;mov   word [HDSM_TBL_VEC],HD2_DPT      ; PARM TABLE DRIVE 82h
6471                              <1>       ;mov   word [HDSM_TBL_VEC+2],DPT_SEGM
6472 0000515C C705[98890100]4000- <1>       mov    dword [HDSM_TBL_VEC], (DPT_SEGM*16)+HD2_DPT
6472 00005164 0900                <1>
6473                              <1>       ;mov   word [HDSS_TBL_VEC],HD3_DPT      ; PARM TABLE DRIVE 83h
6474                              <1>       ;mov   word [HDSS_TBL_VEC+2],DPT_SEGM
6475 00005166 C705[9C890100]6000- <1>       mov    dword [HDSS_TBL_VEC], (DPT_SEGM*16)+HD3_DPT
6475 0000516E 0900                <1>
6476                              <1>       ;
6477                              <1>       ;;IN   AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
6478                              <1>       ;;;AND AL,0BFH
6479                              <1>       ;;and  al, 3Fh                  ; enable IRQ 14 and IRQ 15
6480                              <1>       ;;;JMP $+2
6481                              <1>       ;;IODELAY
6482                              <1>       ;;OUT  INTB01,AL
6483                              <1>       ;;IODELAY
6484                              <1>       ;;IN   AL,INTA01          ; LET INTERRUPTS PASS THRU TO
6485                              <1>       ;;AND  AL,0FBH            ;  SECOND CHIP
6486                              <1>       ;;;JMP $+2
6487                              <1>       ;;IODELAY
6488                              <1>       ;;OUT  INTA01,AL
6489                              <1>       ;
6490                              <1>       ;STI
6491                              <1>       ;;PUSH DS                 ; MOVE ABS0 POINTER TO
6492                              <1>       ;;POP  ES                 ; EXTRA SEGMENT POINTER
6493                              <1>       ;;;CALL      DDS                ; ESTABLISH DATA SEGMENT
6494                              <1>       ;;MOV  byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
6495                              <1>       ;;MOV  byte [HF_NUM],0           ; ZERO NUMBER OF FIXED DISKS
6496                              <1>       ;;MOV  byte [CONTROL_BYTE],0
6497                              <1>       ;;MOV  byte [PORT_OFF],0   ; ZERO CARD OFFSET
6498                              <1>       ; 20/12/2014 - private code by Erdogan Tan
6499                              <1>                  ; (out of original PC-AT, PC-XT BIOS code)
6500                              <1>       ;mov   si, hd0_type
6501 00005170 BE[D06C0000]       <1>       mov    esi, hd0_type
6502                              <1>       ;mov   cx, 4
6503 00005175 B904000000         <1>       mov    ecx, 4
6504                              <1> hde_l:
6505 0000517A AC                 <1>       lodsb
6506 0000517B 3C80               <1>       cmp    al, 80h                  ; 8?h = existing
6507 0000517D 7206               <1>       jb     short _L4
6508 0000517F FE05[8C890100]     <1>       inc    byte [HF_NUM]       ; + 1 hard (fixed) disk drives
6509                              <1> _L4: ; 26/02/2015
6510 00005185 E2F3               <1>       loop   hde_l
6511                              <1> ;_L4:                           ; 0 <= [HF_NUM] =< 4
6512                              <1> ;L4:
6513                              <1>       ;
6514                              <1>       ;; 31/12/2014 - cancel controller diagnostics here
6515                              <1>       ;;;mov       cx, 3  ; 26/12/2014 (Award BIOS 1999)
6516                              <1>       ;;mov  cl, 3
6517                              <1>       ;;
6518                              <1>       ;;MOV  DL,80H             ; CHECK THE CONTROLLER
6519                              <1> ;;hdc_dl:
6520                              <1>       ;;MOV  AH,14H             ; USE CONTROLLER DIAGNOSTIC COMMAND
6521                              <1>       ;;INT  13H                ; CALL BIOS WITH DIAGNOSTIC COMMAND
6522                              <1>       ;;;JC  short CTL_ERRX            ; DISPLAY ERROR MESSAGE IF BAD RETURN
6523                              <1>       ;;;jc  short POD_DONE ;22/12/2014
6524                              <1>       ;;jnc  short hdc_reset0
6525                              <1>       ;;loop hdc_dl
6526                              <1>       ;;; 27/12/2014
6527                              <1>       ;;stc
6528                              <1>       ;;retn
6529                              <1>       ;
6530                              <1> ;;hdc_reset0:
6531                              <1>       ; 18/01/2015
6532 00005187 8A0D[8C890100]     <1>       mov    cl, [HF_NUM]
6533 0000518D 20C9               <1>       and    cl, cl
6534 0000518F 740D               <1>       jz     short POD_DONE
6535                              <1>       ;
6536 00005191 B27F               <1>       mov    dl, 7Fh
6537                              <1> hdc_reset1:
6538 00005193 FEC2               <1>       inc    dl
6539                              <1>       ;; 31/12/2015
6540                              <1>       ;;push dx
6541                              <1>       ;;push cx
6542                              <1>       ;;push ds
6543                              <1>       ;;sub  ax, ax
6544                              <1>       ;;mov  ds, ax
6545                              <1>       ;;MOV  AX, [TIMER_LOW]           ; GET START TIMER COUNTS
6546                              <1>       ;;pop  ds
6547                              <1>       ;;MOV  BX,AX
6548                              <1>       ;;ADD  AX,6*182           ; 60 SECONDS* 18.2
6549                              <1>       ;;MOV  CX,AX
6550                              <1>       ;;mov  word [wait_count], 0      ; 22/12/2014 (reset wait counter)
6551                              <1>       ;;
6552                              <1>       ;; 31/12/2014 - cancel HD_RESET_1
6553                              <1>       ;;CALL HD_RESET_1         ; SET UP DRIVE 0, (1,2,3)
6554                              <1>       ;;pop  cx
6555                              <1>       ;;pop  dx
6556                              <1>       ;;
6557                              <1>       ; 18/01/2015
6558 00005195 B40D               <1>       mov    ah, 0Dh ; ALTERNATE RESET
6559                              <1>       ;int   13h
6560 00005197 E8ED000000         <1>       call   int13h
6561 0000519C E2F5               <1>       loop   hdc_reset1
6562                              <1>       ;clc   ; 29/05/2016
6563                              <1> POD_DONE:
6564 0000519E C3                 <1>       RETn
6565                              <1>
6566                              <1> ;;-----     POD_ERROR
6567                              <1>
6568                              <1> ;;CTL_ERRX:
6569                              <1> ;     ;MOV   SI,OFFSET F1782     ; CONTROLLER ERROR
6570                              <1> ;     ;CALL  SET_FAIL            ; DO NOT IPL FROM DISK
6571                              <1> ;     ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
```

```
6572                            <1> ;       ;JMP   short POD_DONE
6573                            <1>
6574                            <1> ;;HD_RESET_1:
6575                            <1> ;;      ;PUSH  BX                   ; SAVE TIMER LIMITS
6576                            <1> ;;      ;PUSH  CX
6577                            <1> ;;RES_1: MOV AH,09H                 ; SET DRIVE PARAMETERS
6578                            <1> ;;      INT    13H
6579                            <1> ;;      JC     short RES_2
6580                            <1> ;;      MOV    AH,11H               ; RECALIBRATE DRIVE
6581                            <1> ;;      INT    13H
6582                            <1> ;;      JNC    short RES_CK         ; DRIVE OK
6583                            <1> ;;RES_2: ;CALL     POD_TCHK              ; CHECK TIME OUT
6584                            <1> ;;      cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
6585                            <1> ;;                                  ; (30 seconds)
6586                            <1> ;;      ;cmc
6587                            <1> ;;      ;JNC   short RES_1
6588                            <1> ;;      jb     short RES_1
6589                            <1> ;;;RES_FL: ;MOV    SI,OFFSET F1781      ; INDICATE DISK 1 FAILURE;
6590                            <1> ;;      ;TEST  DL,1
6591                            <1> ;;      ;JNZ   RES_E1
6592                            <1> ;;      ;MOV   SI,OFFSET F1780      ; INDICATE DISK 0 FAILURE
6593                            <1> ;;      ;CALL  SET_FAIL             ; DO NOT TRY TO IPL DISK 0
6594                            <1> ;;      ;JMP   SHORT RES_E1
6595                            <1> ;;RES_ER: ; 22/12/2014
6596                            <1> ;;RES_OK:
6597                            <1> ;;      ;POP   CX                   ; RESTORE TIMER LIMITS
6598                            <1> ;;      ;POP   BX
6599                            <1> ;;      RETn
6600                            <1> ;;
6601                            <1> ;;RES_RS: MOV      AH,00H               ; RESET THE DRIVE
6602                            <1> ;;      INT    13H
6603                            <1> ;;RES_CK: MOV      AH,08H               ; GET MAX CYLINDER,HEAD,SECTOR
6604                            <1> ;;      MOV    BL,DL                ; SAVE DRIVE CODE
6605                            <1> ;;      INT    13H
6606                            <1> ;;      JC     short RES_ER
6607                            <1> ;;      MOV    [NEC_STATUS],CX      ; SAVE MAX CYLINDER, SECTOR
6608                            <1> ;;      MOV    DL,BL                ; RESTORE DRIVE CODE
6609                            <1> ;;RES_3: MOV AX,0401H               ; VERIFY THE LAST SECTOR
6610                            <1> ;;      INT    13H
6611                            <1> ;;      JNC    short RES_OK         ; VERIFY OK
6612                            <1> ;;      CMP    AH,BAD_SECTOR        ; OK ALSO IF JUST ID READ
6613                            <1> ;;      JE     short RES_OK
6614                            <1> ;;      CMP    AH,DATA_CORRECTED
6615                            <1> ;;      JE     short RES_OK
6616                            <1> ;;      CMP    AH,BAD_ECC
6617                            <1> ;;      JE     short RES_OK
6618                            <1> ;;      ;CALL  POD_TCHK             ; CHECK FOR TIME OUT
6619                            <1> ;;      cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
6620                            <1> ;;                                  ; (60 seconds)
6621                            <1> ;;      cmc
6622                            <1> ;;      JC     short RES_ER         ; FAILED
6623                            <1> ;;      MOV    CX,[NEC_STATUS]      ; GET SECTOR ADDRESS, AND CYLINDER
6624                            <1> ;;      MOV    AL,CL                ; SEPARATE OUT SECTOR NUMBER
6625                            <1> ;;      AND    AL,3FH
6626                            <1> ;;      DEC    AL                   ; TRY PREVIOUS ONE
6627                            <1> ;;      JZ     short RES_RS         ; WE'VE TRIED ALL SECTORS ON TRACK
6628                            <1> ;;      AND    CL,0C0H              ; KEEP CYLINDER BITS
6629                            <1> ;;      OR     CL,AL                ; MERGE SECTOR WITH CYLINDER BITS
6630                            <1> ;;      MOV    [NEC_STATUS],CX      ; SAVE CYLINDER, NEW SECTOR NUMBER
6631                            <1> ;;      JMP    short RES_3          ; TRY AGAIN
6632                            <1> ;;;RES_ER: MOV     SI,OFFSET F1791      ; INDICATE DISK 1 ERROR
6633                            <1> ;;      ;TEST  DL,1
6634                            <1> ;;      ;JNZ   short RES_E1
6635                            <1> ;;      ;MOV   SI,OFFSET F1790      ; INDICATE DISK 0 ERROR
6636                            <1> ;;;RES_E1:
6637                            <1> ;;      ;CALL  E_MSG                ; DISPLAY ERROR AND SET (BP) ERROR FLAG
6638                            <1> ;;;RES_OK:
6639                            <1> ;;      ;POP   CX                   ; RESTORE TIMER LIMITS
6640                            <1> ;;      ;POP   BX
6641                            <1> ;;      ;RETn
6642                            <1> ;
6643                            <1> ;;SET_FAIL:
6644                            <1> ;       ;MOV   AX,X*(CMOS_DIAG+NMI)       ; GET CMOS ERROR BYTE
6645                            <1> ;       ;CALL  CMOS_READ
6646                            <1> ;       ;OR    AL,HF_FAIL           ; SET DO NOT IPL FROM DISK FLAG
6647                            <1> ;       ;XCHG  AH,AL                ; SAVE IT
6648                            <1> ;       ;CALL  CMOS_WRITE           ; PUT IT OUT
6649                            <1> ;       ;RETn
6650                            <1> ;
6651                            <1> ;;POD_TCHK:                         ; CHECK FOR 30 SECOND TIME OUT
6652                            <1> ;       ;POP   AX                   ; SAVE RETURN
6653                            <1> ;       ;POP   CX                   ; GET TIME OUT LIMITS
6654                            <1> ;       ;POP   BX
6655                            <1> ;       ;PUSH  BX                   ; AND SAVE THEM AGAIN
6656                            <1> ;       ;PUSH  CX
6657                            <1> ;       ;PUSH  AX
6658                            <1> ;       ;push  ds
6659                            <1> ;       ;xor   ax, ax
6660                            <1> ;       ;mov   ds, ax               ; RESTORE RETURN
6661                            <1> ;       ;MOV   AX, [TIMER_LOW]           ; AX = CURRENT TIME
6662                            <1> ;       ;                           ; BX = START TIME
6663                            <1> ;       ;                           ; CX = END TIME
6664                            <1> ;       ;pop   ds
6665                            <1> ;       ;CMP   BX,CX
6666                            <1> ;       ;JB    short TCHK1          ; START < END
6667                            <1> ;       ;CMP   BX,AX
6668                            <1> ;       ;JB    short TCHKG          ; END < START < CURRENT
6669                            <1> ;       ;JMP   SHORT TCHK2          ; END, CURRENT < START
6670                            <1> ;;TCHK1: CMP AX,BX
6671                            <1> ;;      JB     short TCHKNG         ; CURRENT < START < END
6672                            <1> ;;TCHK2: CMP AX,CX
6673                            <1> ;;      JB     short TCHKG          ; START < CURRENT < END
6674                            <1> ;;                                  ; OR CURRENT < END < START
6675                            <1> ;;TCHKNG: STC                            ; CARRY SET INDICATES TIME OUT
6676                            <1> ;;      RETn
```

```
                                <1> ;;TCHKG: CLC                    ; INDICATE STILL TIME
                                <1> ;;    RETn
                                <1> ;;
                                <1> ;;int_13h:
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     FIXED DISK BIOS ENTRY POINT      :
                                <1> ;----------------------------------------
                                <1>
                                <1> ; 15/01/2017
                                <1> ; 14/01/2017
                                <1> ; 07/01/2017
                                <1> ; 02/01/2017
                                <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
0000519F 80642408FE             <1>       and   byte [esp+8], 11111110b  ; clear carry bit of eflags register
                                <1>       ; 16/05/2016
000051A4 1E                     <1>       push  ds
000051A5 53                     <1>       push  ebx ; user's buffer address (virtual)
000051A6 66BB1000               <1>       mov   bx, KDATA ; System (Kernel's) data segment
000051AA 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>
000051AC 8F05[7C950100]         <1>       pop   dword [user_buffer] ; 01/06/2016
                                <1>
000051B2 C605[B68E0100]00       <1>       mov   byte [scount], 0 ; sector count for transfer
000051B9 80FC03                 <1>       cmp   ah, 03h ; chs write
000051BC 7744                   <1>       ja    short int33h_2
000051BE 7407                   <1>       je    short int33h_0
000051C0 80FC02                 <1>       cmp   ah, 02h ; chs read
000051C3 726A                   <1>       jb    short int33h_5
000051C5 EB63                   <1>       jmp   short int33h_4
                                <1> int33h_0:
                                <1>       ; transfer user's buffer content to sector buffer
000051C7 51                     <1>       push  ecx
000051C8 0FB6C8                 <1>       movzx ecx, al
                                <1> int33h_1:
000051CB 56                     <1>       push  esi
000051CC 8B35[7C950100]         <1>       mov   esi, [user_buffer]
                                <1>       ; esi = user's buffer address (virtual, ebx)
000051D2 57                     <1>       push  edi
000051D3 06                     <1>       push  es
000051D4 50                     <1>       push  eax
000051D5 66B81000               <1>       mov   ax, KDATA
000051D9 8EC0                   <1>       mov   es, ax
000051DB BF00000700             <1>       mov   edi, Cluster_Buffer
000051E0 C1E109                 <1>       shl   ecx, 9 ; * 512
000051E3 E886C80000             <1>       call  transfer_from_user_buffer
000051E8 58                     <1>       pop   eax
000051E9 07                     <1>       pop   es
000051EA 5F                     <1>       pop   edi
000051EB 5E                     <1>       pop   esi
000051EC 59                     <1>       pop   ecx
000051ED 7340                   <1>       jnc   short int33h_5
000051EF 8B1D[7C950100]         <1>       mov   ebx, [user_buffer] ; 01/06/2016
000051F5 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
000051F6 804C240801             <1>       or    byte [esp+8], 1  ; set carry bit of eflags register
                                <1>       ; [esp+12] = ESP (user)
                                <1>       ; [esp+16] = SS (User)
000051FB B8FF000000             <1>       mov   eax, 0FFh ; Unknown error !?
                                <1>       ;iretd
00005200 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:
```

```
6782 00005202 80FC05              <1>       cmp   ah, 05h ; format track
6783 00005205 770A                <1>       ja    short int33h_3
6784 00005207 7226                <1>       jb    short int33h_5
6785 00005209 51                  <1>       push  ecx
6786 0000520A B901000000          <1>       mov   ecx, 1
6787 0000520F EBBA                <1>       jmp   short int33h_1
6788                              <1> int33h_3:
6789 00005211 80FC1C              <1>       cmp   ah, 1Ch ; LBA write
6790 00005214 7719                <1>       ja    short int33h_5
6791 00005216 74AF                <1>       je    short int33h_0
6792 00005218 80FC1B              <1>       cmp   ah, 1Bh ; LBA read
6793 0000521B 740D                <1>       je    short int33h_4
6794 0000521D 80FC08              <1>       cmp   ah, 08h ; get disk parameters
6795 00005220 750D                <1>       jne   short int33h_5
6796                              <1>       ; 01/06/2016
6797 00005222 8B1D[7C950100]      <1>       mov   ebx, [user_buffer] ; user's buffer address
6798 00005228 EB0A                <1>       jmp   short int33h_6
6799                              <1> int33h_4:
6800 0000522A A2[B68E0100]        <1>       mov   byte [scount], al ; <= 128 sectors
6801                              <1> int33h_5:
6802 0000522F BB00000700          <1>       mov   ebx, Cluster_Buffer ; max. 65536 bytes
6803                              <1>                                 ; buf. addr: 70000h
6804                              <1>       ;mov  byte [ClusterBuffer_Valid], 0
6805                              <1> int33h_6:
6806 00005234 1F                  <1>       pop   ds
6807 00005235 9C                  <1>       pushfd
6808 00005236 0E                  <1>       push  cs
6809 00005237 E856000000          <1>       call  DISK_IO
6810 0000523C 2E8B1D[7C950100]    <1>       mov   ebx, [CS:user_buffer] ; 01/06/2016
6811 00005243 723D                <1>       jc    short int33h_9
6812                              <1>       ;
6813 00005245 2E803D[B68E0100]00  <1>       cmp   byte [CS:scount], 0
6814 0000524D 762C                <1>       jna   short int33h_7
6815                              <1>       ; transfer sector buffer content to user's buffer
6816 0000524F 06                  <1>       push  es
6817 00005250 1E                  <1>       push  ds
6818 00005251 50                  <1>       push  eax
6819 00005252 66B81000            <1>       mov   ax, KDATA
6820 00005256 8ED8                <1>       mov   ds, ax
6821 00005258 8EC0                <1>       mov   es, ax
6822 0000525A 51                  <1>       push  ecx
6823 0000525B 56                  <1>       push  esi
6824 0000525C 57                  <1>       push  edi
6825 0000525D 0FB60D[B68E0100]    <1>       movzx ecx, byte [scount]
6826 00005264 C1E109              <1>       shl   ecx, 9 ; * 512 bytes
6827 00005267 89DF                <1>       mov   edi, ebx ; user's buffer address
6828 00005269 BE00000700          <1>       mov   esi, Cluster_Buffer
6829 0000526E E8B1C70000          <1>       call  transfer_to_user_buffer
6830 00005273 5F                  <1>       pop   edi
6831 00005274 5E                  <1>       pop   esi
6832 00005275 59                  <1>       pop   ecx
6833 00005276 58                  <1>       pop   eax
6834 00005277 1F                  <1>       pop   ds
6835 00005278 07                  <1>       pop   es
6836 00005279 7202                <1>       jc    short int33h_8
6837                              <1> int33h_7:
6838 0000527B FA                  <1>       cli
6839                              <1>       ;;15/01/2017
6840                              <1>       ;;mov byte [ss:intflg], 0 ; 07/01/2017
6841                              <1>       ; cf = 0  ; use eflags which is in stack
6842 0000527C CF                  <1>       iretd
6843                              <1> int33h_8:
6844 0000527D B8FF000000          <1>       mov   eax, 0FFh ; Unknown error !?
6845                              <1> int33h_9:
6846                              <1>       ; cf = 1
6847                              <1>
6848                              <1>       ; (*) 29/05/2016
6849                              <1>       ; (*) retf 4 ; skip eflags on stack
6850                              <1>       ; Note: This 'retf 4' was wrong, -it was causing
6851                              <1>       ;       to stack errors in ring 3-
6852                              <1>       ;     POP sequence of 'retf 4' is as
6853                              <1>       ;       "eip, cs, eflags, esp, ss, +4 bytes"
6854                              <1>       ;       it is not as "eip, cs, +4 bytes, esp, ss" !
6855                              <1>
6856                              <1>       ; 29/05/2016 -set carry flag on stack-
6857 00005282 804C240801          <1>       or    byte [esp+8], 1  ; set carry bit of eflags register
6858                              <1>       ;iretd
6859 00005287 EBF2                <1>       jmp   short int33h_7 ; 07/01/2017
6860                              <1>
6861                              <1> ; 11/04/2021
6862                              <1> int13h: ; 21/02/2015
6863 00005289 F8                  <1>       clc ; 11/04/2021
6864 0000528A 9C                  <1>       pushfd
6865 0000528B 0E                  <1>       push  cs
6866 0000528C E801000000          <1>       call  DISK_IO
6867 00005291 C3                  <1>       retn
6868                              <1>
6869                              <1> ; 30/08/2020
6870                              <1> ; 09/12/2017
6871                              <1> ; 29/05/2016
6872                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
6873                              <1>
6874                              <1> DISK_IO:
6875 00005292 80FA80              <1>       CMP   DL,80H              ; TEST FOR FIXED DISK DRIVE
6876                              <1>       ;JAE  short A1            ; YES, HANDLE HERE
6877                              <1>       ;;;INT 40H                ; DISKETTE HANDLER
6878                              <1>       ;;call int40h
6879                              <1>       ;jb   DISKETTE_IO_1
6880                              <1> ;RET_2:
6881                              <1>       ;RETf 2                   ; BACK TO CALLER
6882                              <1> ;     retf  4
6883                              <1>       ; 11/04/2021
6884 00005295 7305                <1>       jnb   short A1
6885 00005297 E940F0FFFF          <1>       jmp   DISKETTE_IO_1
6886                              <1> A1:
```

```
6887 0000529C FB                <1>       STI                         ; ENABLE INTERRUPTS
6888                            <1>       ;; 04/01/2015
6889                            <1>       ;;OR   AH,AH
6890                            <1>       ;;JNZ  short A2
6891                            <1>       ;;INT  40H                  ; RESET NEC WHEN AH=0
6892                            <1>       ;;SUB  AH,AH
6893 0000529D 80FA83            <1>       CMP    DL,(80H + S_MAX_FILE - 1)
6894                            <1>       ;JA    short RET_2
6895 000052A0 7614              <1>       jna    short _A0
6896                            <1>       ; 29/05/2016
6897 000052A2 1E                <1>       push   ds
6898                            <1>       ;push  ax
6899                            <1>       ; 11/04/2021
6900 000052A3 50                <1>       push   eax
6901 000052A4 66B81000          <1>       mov    ax, KDATA
6902 000052A8 8ED8              <1>       mov    ds, ax
6903                            <1>       ;pop   ax
6904                            <1>       ; 11/04/2021
6905 000052AA 58                <1>       pop    eax
6906 000052AB B4AA              <1>       mov     ah, 0AAh        ; Hard disk drive not ready !
6907                            <1>                               ; (Programmer's guide to AMIBIOS, 1992)
6908 000052AD 8825[8B890100]    <1>       mov     byte [DISK_STATUS1], ah
6909 000052B3 1F                <1>       pop    ds
6910 000052B4 EB38              <1>       jmp    short RET_2
6911                            <1> _A0:
6912                            <1>       ; 18/01/2015
6913 000052B6 08E4              <1>       or     ah,ah
6914 000052B8 743A              <1>       jz     short A4
6915 000052BA 80FC0D            <1>       cmp    ah, 0Dh      ; Alternate reset
6916 000052BD 7504              <1>       jne    short A2
6917 000052BF 28E4              <1>       sub    ah,ah  ; Reset
6918 000052C1 EB31              <1>       jmp    short A4
6919                            <1> A2:
6920 000052C3 80FC08            <1>       CMP    AH,08H              ; GET PARAMETERS IS A SPECIAL CASE
6921                            <1>       ;JNZ   short A3
6922                            <1>         ;JMP    GET_PARM_N
6923 000052C6 0F8427030000      <1>       je     GET_PARM_N
6924 000052CC 80FC15            <1> A3:   CMP    AH,15H              ; READ DASD TYPE IS ALSO
6925                            <1>       ;JNZ   short A4
6926                            <1>         ;JMP    READ_DASD_TYPE
6927 000052CF 0F84D0020000      <1>         je      READ_DASD_TYPE
6928                            <1>       ; 02/02/2015
6929 000052D5 80FC1D            <1>       cmp    ah, 1Dh                  ;(Temporary for Retro UNIX 386 v1)
6930                            <1>       ; 12/01/2015
6931 000052D8 F5                <1>       cmc
6932 000052D9 7319              <1>       jnc    short A4
6933                            <1> int33h_bad_cmd:
6934                            <1>       ; 16/05/2016
6935                            <1>       ; 30/01/2015
6936                            <1>       ; 29/05/2016
6937 000052DB 1E                <1>       push   ds
6938 000052DC 6650              <1>       push   ax
6939 000052DE 66B81000          <1>       mov    ax, KDATA
6940 000052E2 8ED8              <1>       mov    ds, ax
6941 000052E4 6658              <1>       pop    ax
6942 000052E6 B401              <1>       mov    ah, BAD_CMD
6943 000052E8 8825[8B890100]    <1>       mov     [DISK_STATUS1], ah ; BAD_CMD  ; COMMAND ERROR
6944                            <1>         ;jmp short RET_2
6945                            <1> RET_2:
6946                            <1>       ; (*) 29/05/2016
6947                            <1>       ; (*) retf 4
6948 000052EE 804C240801        <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
6949 000052F3 CF                <1>       iretd
6950                            <1> A4:                             ; SAVE REGISTERS DURING OPERATION
6951 000052F4 C8080000          <1>       ENTER  8,0                ; SAVE (BP) AND MAKE ROOM FOR @CMD_BLOCK
6952 000052F8 53                <1>       PUSH   eBX                ;  IN THE STACK, THE COMMAND BLOCK IS:
6953 000052F9 51                <1>       PUSH   eCX                ;   @CMD_BLOCK == BYTE PTR [BP]-8
6954 000052FA 52                <1>       PUSH   eDX
6955 000052FB 1E                <1>       PUSH   DS
6956 000052FC 06                <1>       PUSH   ES
6957 000052FD 56                <1>       PUSH   eSI
6958 000052FE 57                <1>       PUSH   eDI
6959                            <1>       ;;04/01/2015
6960                            <1>       ;;OR   AH,AH              ; CHECK FOR RESET
6961                            <1>       ;;JNZ  short A5
6962                            <1>       ;;MOV  DL,80H             ; FORCE DRIVE 80 FOR RESET
6963                            <1> ;;A5:
6964                            <1>       ;push  cs
6965                            <1>       ;pop   ds
6966                            <1>       ; 21/02/2015
6967                            <1>       ;push  ax
6968                            <1>       ; 11/04/2021
6969 000052FF 50                <1>       push   eax
6970 00005300 66B81000          <1>       mov    ax, KDATA
6971 00005304 8ED8              <1>       mov    ds, ax
6972 00005306 8EC0              <1>       mov    es, ax
6973                            <1>       ;pop   ax
6974                            <1>       ; 11/04/2021
6975 00005308 58                <1>       pop    eax
6976 00005309 E88D000000        <1>       CALL   DISK_IO_CONT       ; PERFORM THE OPERATION
6977                            <1>       ;;CALL DDS                ; ESTABLISH SEGMENT
6978 0000530E 8A25[8B890100]    <1>       MOV    AH,[DISK_STATUS1]  ; GET STATUS FROM OPERATION
6979                            <1>       ;(*) CMP AH,1             ; SET THE CARRY FLAG TO INDICATE
6980                            <1>       ;(*) CMC                  ; SUCCESS OR FAILURE
6981 00005314 5F                <1>       POP    eDI                ; RESTORE REGISTERS
6982 00005315 5E                <1>       POP    eSI
6983 00005316 07                <1>       POP   ES
6984 00005317 1F                <1>       POP   DS
6985 00005318 5A                <1>       POP    eDX
6986 00005319 59                <1>       POP    eCX
6987 0000531A 5B                <1>       POP    eBX
6988 0000531B C9                <1>       LEAVE                     ; ADJUST (SP) AND RESTORE (BP)
6989                            <1>       ;RETf  2                  ; THROW AWAY SAVED FLAGS
6990                            <1>       ; (*) 29/05/2016
6991                            <1>       ; (*) retf 4
```

```
6992 0000531C 80FC01              <1>       cmp    ah, 1
6993 0000531F 7205                <1>       jc     short _A5
6994 00005321 804C240801          <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
6995                              <1> _A5:
6996 00005326 CF                  <1>       iretd
6997                              <1>
6998                              <1> ; 21/02/2015
6999                              <1> ;       dw --> dd
7000                              <1> D1:                             ; FUNCTION TRANSFER TABLE
7001 00005327 [E1540000]          <1>       dd     DISK_RESET         ; 000H
7002 0000532B [58550000]          <1>       dd     RETURN_STATUS      ; 001H
7003 0000532F [65550000]          <1>       dd     DISK_READ          ; 002H
7004 00005333 [6E550000]          <1>       dd     DISK_WRITE         ; 003H
7005 00005337 [77550000]          <1>       dd     DISK_VERF          ; 004H
7006 0000533B [8F550000]          <1>       dd     FMT_TRK            ; 005H
7007 0000533F [D7540000]          <1>       dd     BAD_COMMAND        ; 006H FORMAT BAD SECTORS
7008 00005343 [D7540000]          <1>       dd     BAD_COMMAND        ; 007H FORMAT DRIVE
7009 00005347 [D7540000]          <1>       dd     BAD_COMMAND        ; 008H RETURN PARAMETERS
7010 0000534B [7A560000]          <1>       dd     INIT_DRV           ; 009H
7011 0000534F [D9560000]          <1>       dd     RD_LONG            ; 00AH
7012 00005353 [E2560000]          <1>       dd     WR_LONG            ; 00BH
7013 00005357 [EB560000]          <1>       dd     DISK_SEEK          ; 00CH
7014 0000535B [E1540000]          <1>       dd     DISK_RESET         ; 00DH
7015 0000535F [D7540000]          <1>       dd     BAD_COMMAND        ; 00EH READ BUFFER
7016 00005363 [D7540000]          <1>       dd     BAD_COMMAND        ; 00FH WRITE BUFFER
7017 00005367 [13570000]          <1>       dd     TST_RDY            ; 010H
7018 0000536B [37570000]          <1>       dd     HDISK_RECAL        ; 011H
7019 0000536F [D7540000]          <1>       dd     BAD_COMMAND        ; 012H MEMORY DIAGNOSTIC
7020 00005373 [D7540000]          <1>       dd     BAD_COMMAND        ; 013H DRIVE DIAGNOSTIC
7021 00005377 [6D570000]          <1>       dd     CTLR_DIAGNOSTIC    ; 014H CONTROLLER DIAGNOSTIC
7022                              <1>       ; 02/02/2015 (Temporary - Retro UNIX 386 v1 - DISK I/O test)
7023 0000537B [D7540000]          <1>       dd     BAD_COMMAND        ; 015h
7024 0000537F [D7540000]          <1>       dd     BAD_COMMAND        ; 016h
7025 00005383 [D7540000]          <1>       dd     BAD_COMMAND        ; 017h
7026 00005387 [D7540000]          <1>       dd     BAD_COMMAND        ; 018h
7027 0000538B [D7540000]          <1>       dd     BAD_COMMAND        ; 019h
7028 0000538F [D7540000]          <1>       dd     BAD_COMMAND        ; 01Ah
7029 00005393 [65550000]          <1>       dd     DISK_READ          ; 01Bh ; LBA read
7030 00005397 [6E550000]          <1>       dd     DISK_WRITE         ; 01Ch ; LBA write
7031                              <1> D1L  EQU    $ - D1
7032                              <1>
7033                              <1> DISK_IO_CONT:
7034                              <1>       ;;CALL DDS                 ; ESTABLISH SEGMENT
7035 0000539B 80FC01              <1>       CMP    AH,01H              ; RETURN STATUS
7036                              <1>       ;;JNZ  short SU0
7037                              <1>       ;;JMP  RETURN_STATUS
7038                              <1>       ;je    RETURN_STATUS
7039                              <1>       ; 11/04/2021
7040 0000539E 7505                <1>       jne    short SU0
7041 000053A0 E9B3010000          <1>       jmp    RETURN_STATUS
7042                              <1> SU0:
7043 000053A5 C605[8B890100]00    <1>       MOV    byte [DISK_STATUS1],0 ; RESET THE STATUS INDICATOR
7044                              <1>       ;;PUSH BX                  ; SAVE DATA ADDRESS
7045                              <1>       ;mov   si, bx ;; 14/02/2015
7046 000053AC 89DE                <1>       mov    esi, ebx ; 21/02/2015
7047 000053AE 8A1D[8C890100]      <1>       MOV    BL,[HF_NUM]         ; GET NUMBER OF DRIVES
7048                              <1>       ;; 04/01/2015
7049                              <1>       ;;PUSH AX
7050 000053B4 80E27F              <1>       AND    DL,7FH              ; GET DRIVE AS 0 OR 1
7051                              <1>                                  ; (get drive number as 0 to 3)
7052 000053B7 38D3                <1>       CMP    BL,DL
7053                              <1>         ;;JBE   BAD_COMMAND_POP         ; INVALID DRIVE
7054 000053B9 0F8618010000        <1>         jbe     BAD_COMMAND ;; 14/02/2015
7055                              <1>         ;
7056                              <1>       ;;03/01/2015
7057 000053BF 29DB                <1>       sub    ebx, ebx
7058 000053C1 88D3                <1>       mov    bl, dl
7059                              <1>       ;sub   bh, bh
7060 000053C3 883D[A0890100]      <1>       mov    [LBAMode], bh       ; 0
7061                              <1>       ;;test byte [bx+hd0_type], 1     ; LBA ready ?
7062                              <1>       ;test  byte [ebx+hd0_type], 1
7063                              <1>       ;jz    short su1            ; no
7064                              <1>       ;inc   byte [LBAMode]
7065                              <1> ;su1:
7066                              <1>       ; 11/04/2021 (32 bit push/pop)
7067                              <1>       ; 21/02/2015 (32 bit modification)
7068                              <1>       ;04/01/2015
7069 000053C9 50                  <1>       push   Eax ; ***
7070                              <1>       ;PUSH  ES ; **
7071 000053CA 52                  <1>       PUSH   eDX ; *
7072 000053CB 50                  <1>       push   Eax ; ****
7073 000053CC E87F060000          <1>       CALL   GET_VEC             ; GET DISK PARAMETERS
7074                              <1>       ; 02/02/2015
7075                              <1>       ;mov   ax, [ES:BX+16] ; I/O port base address (1F0h, 170h)
7076 000053D1 668B4310            <1>       mov    ax, [ebx+16]
7077 000053D5 66A3[C06C0000]      <1>       mov    [HF_PORT], ax
7078                              <1>       ;mov   dx, [ES:BX+18] ; control port address (3F6h, 376h)
7079 000053DB 668B5312            <1>       mov    dx, [ebx+18]
7080 000053DF 668915[C26C0000]    <1>       mov    [HF_REG_PORT], dx
7081                              <1>       ;mov   al, [ES:BX+20] ; head register upper nibble (A0h,B0h,E0h,F0h)
7082 000053E6 8A4314              <1>       mov    al, [ebx+20]
7083                              <1>       ; 23/02/2015
7084 000053E9 A840                <1>       test   al, 40h       ; LBA bit (bit 6)
7085 000053EB 7406                <1>       jz     short su1
7086 000053ED FE05[A0890100]      <1>       inc    byte [LBAMode] ; 1
7087                              <1> su1:
7088 000053F3 C0E804              <1>       shr    al, 4
7089 000053F6 2401                <1>       and    al, 1
7090 000053F8 A2[C46C0000]        <1>       mov    [hf_m_s], al
7091                              <1>       ;
7092                              <1>       ; 03/01/2015
7093                              <1>       ;MOV   AL,byte [ES:BX+8]   ; GET CONTROL BYTE MODIFIER
7094 000053FD 8A4308              <1>       mov    al, [ebx+8]
7095                              <1>       ;MOV   DX,[HF_REG_PORT]    ; Device Control register
7096 00005400 EE                  <1>       OUT    DX,AL               ; SET EXTRA HEAD OPTION
```

```
7097                                    <1>                                     ; Control Byte:  (= 08h, here)
7098                                    <1>                                     ; bit 0 - 0
7099                                    <1>                                     ; bit 1 - nIEN (1 = disable irq)
7100                                    <1>                                     ; bit 2 - SRST (software RESET)
7101                                    <1>                                     ; bit 3 - use extra heads (8 to 15)
7102                                    <1>                                     ;         -always set to 1-
7103                                    <1>                                     ; (bits 3 to 7 are reserved
7104                                    <1>                                     ;          for ATA devices)
7105 00005401 8A25[8D890100]           <1>         MOV     AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
7106 00005407 80E4C0                   <1>         AND     AH,0C0H             ; CONTROL BYTE
7107 0000540A 08C4                     <1>         OR      AH,AL
7108 0000540C 8825[8D890100]           <1>         MOV     [CONTROL_BYTE],AH
7109                                    <1>         ; 11/04/2021 (32 bit push/pop)
7110                                    <1>         ; 04/01/2015
7111 00005412 58                       <1>         pop     Eax ; ****
7112 00005413 5A                       <1>         pop     Edx ; * ;; 14/02/2015
7113 00005414 20E4                     <1>         and     ah, ah ; Reset function ?
7114 00005416 7506                     <1>         jnz     short su2
7115                                    <1>         ;;pop   dx ; * ;; 14/02/2015
7116                                    <1>         ;pop    es ; **
7117 00005418 58                       <1>         pop     Eax ; ***
7118                                    <1>         ;;pop   bx
7119 00005419 E9C3000000               <1>         jmp     DISK_RESET
7120                                    <1> su2:
7121 0000541E 803D[A0890100]00         <1>         cmp     byte [LBAMode], 0
7122 00005425 7660                     <1>         jna     short su3
7123                                    <1>         ;
7124                                    <1>         ; 02/02/2015 (LBA read/write function calls)
7125 00005427 80FC1B                   <1>         cmp     ah, 1Bh
7126 0000542A 720B                     <1>         jb      short lbarw1
7127 0000542C 80FC1C                   <1>         cmp     ah, 1Ch
7128 0000542F 775B                     <1>         ja      short invldfnc
7129                                    <1>         ;;pop   dx ; * ; 14/02/2015
7130                                    <1>         ;mov    ax, cx ; Lower word of LBA address (bits 0-15)
7131 00005431 89C8                     <1>         mov     eax, ecx ; LBA address (21/02/2015)
7132                                    <1>         ;; 14/02/2015
7133 00005433 88D1                     <1>         mov     cl, dl ; 14/02/2015
7134                                    <1>         ;;mov   dx, bx
7135                                    <1>         ;mov    dx, si ; higher word of LBA address (bits 16-23)
7136                                    <1>         ;;mov   bx, di
7137                                    <1>         ;mov    si, di ; Buffer offset
7138 00005435 EB30                     <1>         jmp     short lbarw2
7139                                    <1> lbarw1:
7140                                    <1>         ; convert CHS to LBA
7141                                    <1>         ;
7142                                    <1>         ; LBA calculation - AWARD BIOS - 1999 - AHDSK.ASM
7143                                    <1>         ; LBA = "# of Heads" * Sectors/Track * Cylinder + Head * Sectors/Track
7144                                    <1>         ;       + Sector - 1
7145                                    <1>         ; 11/04/2021 (32 bit push/pop)
7146 00005437 52                       <1>         push    Edx ; * ;; 14/02/2015
7147                                    <1>         ;xor    dh, dh
7148 00005438 31D2                     <1>         xor     edx, edx
7149                                    <1>         ;mov    dl, [ES:BX+14]      ; sectors per track (logical)
7150 0000543A 8A530E                   <1>         mov     dl, [ebx+14]
7151                                    <1>         ;xor    ah, ah
7152 0000543D 31C0                     <1>         xor     eax, eax
7153                                    <1>         ;mov    al, [ES:BX+2]; heads (logical)
7154 0000543F 8A4302                   <1>         mov     al, [ebx+2]
7155 00005442 FEC8                     <1>         dec     al
7156 00005444 6640                     <1>         inc     ax              ; 0 =  256
7157 00005446 66F7E2                   <1>         mul     dx
7158                                    <1>                 ; AX = # of Heads" * Sectors/Track
7159 00005449 6689CA                   <1>         mov     dx, cx
7160                                    <1>         ;and    cx, 3Fh         ; sector  (1 to 63)
7161 0000544C 83E13F                   <1>         and     ecx, 3fh
7162 0000544F 86D6                     <1>         xchg    dl, dh
7163 00005451 C0EE06                   <1>         shr     dh, 6
7164                                    <1>                 ; DX = cylinder (0 to 1023)
7165                                    <1>         ;mul    dx
7166                                    <1>                 ; DX:AX = # of Heads" * Sectors/Track * Cylinder
7167 00005454 F7E2                     <1>         mul     edx
7168 00005456 FEC9                     <1>         dec     cl  ; sector - 1
7169                                    <1>         ;add    ax, cx
7170                                    <1>         ;adc    dx, 0
7171                                    <1>                 ; DX:AX = # of Heads" * Sectors/Track * Cylinder + Sector -1
7172 00005458 01C8                     <1>         add     eax, ecx
7173                                    <1>         ; 11/04/2021 (32 bit push/pop)
7174 0000545A 59                       <1>         pop     Ecx ; * ; ch = head, cl = drive number (zero based)
7175                                    <1>         ;push   dx
7176                                    <1>         ;push   ax
7177 0000545B 50                       <1>         push    eax
7178                                    <1>         ;mov    al, [ES:BX+14]      ; sectors per track (logical)
7179 0000545C 8A430E                   <1>         mov     al, [ebx+14]
7180 0000545F F6E5                     <1>         mul     ch
7181                                    <1>                 ; AX = Head * Sectors/Track
7182 00005461 0FB7C0                   <1>         movzx   eax, ax ; 09/12/2017
7183                                    <1>         ;pop    dx
7184 00005464 5A                       <1>         pop     edx
7185                                    <1>         ;add    ax, dx
7186                                    <1>         ;pop    dx
7187                                    <1>         ;adc    dx, 0 ; add carry bit
7188 00005465 01D0                     <1>         add     eax, edx
7189                                    <1> lbarw2:
7190 00005467 29D2                     <1>         sub     edx, edx ; 21/02/2015
7191 00005469 88CA                     <1>         mov     dl, cl ; 21/02/2015
7192 0000546B C645F800                 <1>         mov   byte [CMD_BLOCK], 0 ; Features Register
7193                                    <1>                                 ; NOTE: Features register (1F1h, 171h)
7194                                    <1>                                 ; is not used for ATA device R/W functions.
7195                                    <1>                                 ; It is old/obsolete 'write precompensation'
7196                                    <1>                                 ; register and error register
7197                                    <1>                                 ; for old ATA/IDE devices.
7198                                    <1>         ; 18/01/2014
7199                                    <1>         ;mov    ch, [hf_m_s] ; Drive 0 (master) or 1 (slave)
7200 0000546F 8A0D[C46C0000]           <1>         mov     cl, [hf_m_s]
7201                                    <1>         ;shl    ch, 4           ; bit 4 (drive bit)
```

```
7202                                    <1>       ;or    ch, 0E0h     ; bit 5 = 1
7203                                    <1>                           ; bit 6 = 1 = LBA mode
7204                                    <1>                           ; bit 7 = 1
7205 00005475 80C90E                    <1>       or     cl, 0Eh ; 1110b
7206                                    <1>       ;and   dh, 0Fh             ; LBA byte 4 (bits 24 to 27)
7207 00005478 25FFFFFF0F                <1>       and    eax, 0FFFFFFFh
7208 0000547D C1E11C                    <1>       shl    ecx, 28 ; 21/02/2015
7209                                    <1>       ;or    dh, ch
7210 00005480 09C8                      <1>       or     eax, ecx
7211                                    <1>       ;;mov  [CMD_BLOCK+2], al ; LBA byte 1 (bits 0 to 7)
7212                                    <1>                              ; (Sector Number Register)
7213                                    <1>       ;;mov  [CMD_BLOCK+3], ah ; LBA byte 2 (bits 8 to 15)
7214                                    <1>                              ; (Cylinder Low Register)
7215                                    <1>       ;mov   [CMD_BLOCK+2], ax ; LBA byte 1, 2
7216                                    <1>       ;mov   [CMD_BLOCK+4], dl ; LBA byte 3 (bits 16 to 23)
7217                                    <1>                              ; (Cylinder High Register)
7218                                    <1>       ;;mov  [CMD_BLOCK+5], dh ; LBA byte 4 (bits 24 to 27)
7219                                    <1>                              ; (Drive/Head Register)
7220                                    <1>
7221                                    <1>       ;mov   [CMD_BLOCK+4], dx ; LBA byte 4, LBA & DEV select bits
7222 00005482 8945FA                    <1>       mov    [CMD_BLOCK+2], eax ; 21/02/2015
7223                                    <1>       ;14/02/2015
7224                                    <1>       ;mov   dl, cl ; Drive number (INIT_DRV)
7225 00005485 EB37                      <1>       jmp    short su4
7226                                    <1> su3:
7227                                    <1>       ; 02/02/2015
7228                                    <1>       ; (Temporary functions 1Bh & 1Ch are not valid for CHS mode)
7229 00005487 80FC14                    <1>       cmp    ah, 14h
7230 0000548A 7603                      <1>       jna    short chsfnc
7231                                    <1> invldfnc:
7232                                    <1>         ; 14/02/2015
7233                                    <1>       ;pop   es ; **
7234                                    <1>       ;pop    ax ; ***
7235                                    <1>       ; 11/04/2021
7236 0000548C 58                        <1>       pop    eax ; ***
7237                                    <1>       ;jmp   short BAD_COMMAND_POP
7238 0000548D EB48                      <1>       jmp   short BAD_COMMAND
7239                                    <1> chsfnc:
7240                                    <1>       ;MOV   AX,[ES:BX+5]        ; GET WRITE PRE-COMPENSATION CYLINDER
7241 0000548F 668B4305                  <1>       mov    ax, [ebx+5]
7242 00005493 66C1E802                  <1>       SHR    AX,2
7243 00005497 8845F8                    <1>       MOV    [CMD_BLOCK],AL
7244                                    <1>       ;;MOV  AL,[ES:BX+8]        ; GET CONTROL BYTE MODIFIER
7245                                    <1>       ;;PUSH DX
7246                                    <1>       ;;MOV  DX,[HF_REG_PORT]
7247                                    <1>       ;;OUT  DX,AL               ; SET EXTRA HEAD OPTION
7248                                    <1>       ;;POP  DX ; *
7249                                    <1>       ;;POP  ES ; **
7250                                    <1>       ;;MOV  AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
7251                                    <1>       ;;AND  AH,0C0H             ; CONTROL BYTE
7252                                    <1>       ;;OR   AH,AL
7253                                    <1>       ;;MOV  [CONTROL_BYTE],AH
7254                                    <1>       ;
7255 0000549A 88C8                      <1>       MOV    AL,CL               ; GET SECTOR NUMBER
7256 0000549C 243F                      <1>       AND    AL,3FH
7257 0000549E 8845FA                    <1>       MOV    [CMD_BLOCK+2],AL
7258 000054A1 886DFB                    <1>       MOV    [CMD_BLOCK+3],CH    ; GET CYLINDER NUMBER
7259 000054A4 88C8                      <1>       MOV    AL,CL
7260 000054A6 C0E806                    <1>       SHR    AL,6
7261 000054A9 8845FC                    <1>       MOV    [CMD_BLOCK+4],AL    ; CYLINDER HIGH ORDER 2 BITS
7262                                    <1>       ;;05/01/2015
7263                                    <1>       ;;MOV  AL,DL               ; DRIVE NUMBER
7264 000054AC A0[C46C0000]              <1>       mov    al, [hf_m_s]
7265 000054B1 C0E004                    <1>       SHL    AL,4
7266 000054B4 80E60F                    <1>       AND    DH,0FH              ; HEAD NUMBER
7267 000054B7 08F0                      <1>       OR     AL,DH
7268                                    <1>       ;OR    AL,80H or 20H
7269 000054B9 0CA0                      <1>       OR     AL,80h+20h          ; ECC AND 512 BYTE SECTORS
7270 000054BB 8845FD                    <1>       MOV    [CMD_BLOCK+5],AL    ; ECC/SIZE/DRIVE/HEAD
7271                                    <1> su4:
7272                                    <1>       ;POP   ES ; **
7273                                    <1>         ;; 14/02/2015
7274                                    <1>         ;;POP   AX
7275                                    <1>         ;;MOV   [CMD_BLOCK+1],AL ; SECTOR COUNT
7276                                    <1>         ;;PUSH  AX
7277                                    <1>         ;;MOV   AL,AH           ; GET INTO LOW BYTE
7278                                    <1>         ;;XOR   AH,AH           ; ZERO HIGH BYTE
7279                                    <1>         ;;SAL   AX,1            ; *2 FOR TABLE LOOKUP
7280                                    <1>       ;pop   ax ; ***
7281                                    <1>         ; 11/04/2021
7282 000054BE 58                        <1>       pop    eax ; ***
7283 000054BF 8845F9                    <1>       mov    [CMD_BLOCK+1], al
7284 000054C2 29DB                      <1>       sub    ebx, ebx
7285 000054C4 88E3                      <1>       mov     bl, ah
7286                                    <1>         ;xor bh, bh
7287                                    <1>         ;sal bx, 1
7288 000054C6 66C1E302                  <1>         sal  bx, 2  ; 32 bit offset (21/02/2015)
7289                                    <1>       ;;MOV   SI,AX             ; PUT INTO SI FOR BRANCH
7290                                    <1>         ;;CMP   AX,D1L          ; TEST WITHIN RANGE
7291                                    <1>         ;;JNB   short BAD_COMMAND_POP
7292                                    <1>         ;cmp bx, D1L
7293 000054CA 83FB74                    <1>       cmp    ebx, D1L
7294 000054CD 7308                      <1>       jnb    short BAD_COMMAND
7295                                    <1>         ;xchg bx, si
7296 000054CF 87DE                      <1>         xchg ebx, esi
7297                                    <1>       ;;;POP AX                 ; RESTORE AX
7298                                    <1>       ;;;POP BX                 ; AND DATA ADDRESS
7299                                    <1>
7300                                    <1>       ;;PUSH CX
7301                                    <1>       ;;PUSH AX                 ; ADJUST ES:BX
7302                                    <1>       ;MOV   CX,BX              ; GET 3 HIGH ORDER NIBBLES OF BX
7303                                    <1>       ;SHR   CX,4
7304                                    <1>       ;MOV   AX,ES
7305                                    <1>       ;ADD   AX,CX
7306                                    <1>       ;MOV   ES,AX
```

```
7307                            <1>       ;AND   BX,000FH           ; ES:BX CHANGED TO ES:000X
7308                            <1>       ;;POP  AX
7309                            <1>       ;;POP  CX
7310                            <1>       ;;JMP  word [CS:SI+D1]
7311                            <1>       ;jmp   word [SI+D1]
7312 000054D1 FFA6[27530000]    <1>       jmp    dword [esi+D1]
7313                            <1> ;;BAD_COMMAND_POP:
7314                            <1> ;;     POP    AX
7315                            <1> ;;     POP    BX
7316                            <1> BAD_COMMAND:
7317 000054D7 C605[8B890100]01  <1>       MOV   byte [DISK_STATUS1],BAD_CMD ; COMMAND ERROR
7318 000054DE B000              <1>       MOV    AL,0
7319 000054E0 C3                <1>       RETn
7320                            <1>
7321                            <1> ;--------------------------------------
7322                            <1> ;     RESET THE DISK SYSTEM  (AH=00H) :
7323                            <1> ;--------------------------------------
7324                            <1>
7325                            <1> ; 18-1-2015 : one controller reset (not other one)
7326                            <1>
7327                            <1> DISK_RESET:
7328 000054E1 FA                <1>       CLI
7329 000054E2 E4A1              <1>       IN     AL,INTB01          ; GET THE MASK REGISTER
7330                            <1>       ;JMP   $+2
7331                            <1>       IODELAY
2968 000054E4 EB00              <2>  jmp short $+2
2969 000054E6 EB00              <2>  jmp short $+2
7332                            <1>       ;AND   AL,0BFH            ; ENABLE FIXED DISK INTERRUPT
7333 000054E8 243F              <1>       and    al,3Fh             ; 22/12/2014 (IRQ 14 & IRQ 15)
7334 000054EA E6A1              <1>       OUT    INTB01,AL
7335 000054EC FB                <1>       STI                       ; START INTERRUPTS
7336                            <1>       ; 14/02/2015
7337 000054ED 6689D7            <1>       mov    di, dx
7338                            <1>       ; 04/01/2015
7339                            <1>       ;xor   di,di
7340                            <1> drst0:
7341 000054F0 B004              <1>       MOV    AL,04H  ; bit 2 - SRST
7342                            <1>       ;MOV   DX,HF_REG_PORT
7343 000054F2 668B15[C26C0000]  <1>       MOV    DX,[HF_REG_PORT]
7344 000054F9 EE                <1>       OUT    DX,AL              ; RESET
7345                            <1> ;     MOV    CX,10              ; DELAY COUNT
7346                            <1> ;DRD: DEC    CX
7347                            <1> ;     JNZ    short DRD          ; WAIT 4.8 MICRO-SEC
7348                            <1>       ;mov   cx,2               ; wait for 30 micro seconds
7349 000054FA B902000000        <1>       mov    ecx, 2 ; 21/02/2015
7350 000054FF E8C5CEFFFF        <1>       call    WAITF             ; (Award Bios 1999 - WAIT_REFRESH,
7351                            <1>                                 ; 40 micro seconds)
7352 00005504 A0[8D890100]      <1>       mov    al,[CONTROL_BYTE]
7353 00005509 240F              <1>       AND    AL,0FH             ; SET HEAD OPTION
7354 0000550B EE                <1>       OUT    DX,AL              ; TURN RESET OFF
7355 0000550C E838040000        <1>       CALL   NOT_BUSY
7356 00005511 7515              <1>       JNZ    short DRERR        ; TIME OUT ON RESET
7357 00005513 668B15[C06C0000]  <1>       MOV    DX,[HF_PORT]
7358 0000551A FEC2              <1>       inc    dl  ; HF_PORT+1
7359                            <1>       ; 02/01/2015 - Award BIOS 1999 - AHDSK.ASM
7360                            <1>       ;mov  cl, 10
7361 0000551C B90A000000        <1>       mov   ecx, 10 ; 21/02/2015
7362                            <1> drst1:
7363 00005521 EC                <1>       IN     AL,DX              ; GET RESET STATUS
7364 00005522 3C01              <1>       CMP    AL,1
7365                            <1>       ; 04/01/2015
7366 00005524 740A              <1>       jz     short drst2
7367                            <1>       ;JNZ   short DRERR        ; BAD RESET STATUS
7368                            <1>              ; Drive/Head Register - bit 4
7369 00005526 E2F9              <1>       loop   drst1
7370                            <1> DRERR:
7371 00005528 C605[8B890100]05  <1>       MOV    byte [DISK_STATUS1],BAD_RESET ; CARD FAILED
7372 0000552F C3                <1>       RETn
7373                            <1> drst2:
7374                            <1>       ; 14/02/2015
7375 00005530 6689FA            <1>       mov    dx,di
7376                            <1> ;drst3:
7377                            <1> ;     ; 05/01/2015
7378                            <1> ;     shl    di,1
7379                            <1> ;     ; 04/01/2015
7380                            <1> ;     mov    ax,[di+hd_cports]
7381                            <1> ;     cmp    ax,[HF_REG_PORT]
7382                            <1> ;     je     short drst4
7383                            <1> ;     mov    [HF_REG_PORT], ax
7384                            <1> ;     ; 03/01/2015
7385                            <1> ;     mov    ax,[di+hd_ports]
7386                            <1> ;       mov     [HF_PORT], ax
7387                            <1> ;     ; 05/01/2014
7388                            <1> ;     shr    di,1
7389                            <1> ;     ; 04/01/2015
7390                            <1> ;     jmp    short drst0  ; reset other controller
7391                            <1> ;drst4:
7392                            <1> ;     ; 05/01/2015
7393                            <1> ;     shr    di,1
7394                            <1> ;     mov    al,[di+hd_dregs]
7395                            <1> ;     and    al,10h ; bit 4 only
7396                            <1> ;     shr    al,4 ; bit 4  -> bit 0
7397                            <1> ;     mov    [hf_m_s], al ; (0 = master, 1 = slave)
7398                            <1>       ;
7399 00005533 A0[C46C0000]      <1>       mov    al, [hf_m_s] ; 18/01/2015
7400 00005538 A801              <1>       test   al,1
7401                            <1> ;     jnz    short drst6
7402 0000553A 7516              <1>       jnz   short drst4
7403 0000553C 8065FDEF          <1>       AND   byte [CMD_BLOCK+5],0EFH ; SET TO DRIVE 0
7404                            <1> ;drst5:
7405                            <1> drst3:
7406 00005540 E835010000        <1>       CALL   INIT_DRV           ; SET MAX HEADS
7407                            <1>       ;mov   dx,di
7408 00005545 E8ED010000        <1>       CALL   HDISK_RECAL        ; RECAL TO RESET SEEK SPEED
7409                            <1>       ; 04/01/2014
```

```
7410                            <1> ;       inc     di
7411                            <1> ;       mov     dx,di
7412                            <1> ;       cmp     dl,[HF_NUM]
7413                            <1> ;       jb      short drst3
7414                            <1> ;DRE:
7415 0000554A C605[8B890100]00  <1>         MOV     byte [DISK_STATUS1],0       ; IGNORE ANY SET UP ERRORS
7416 00005551 C3                <1>         RETn
7417                            <1> ;drst6:
7418                            <1> drst4:          ; Drive/Head Register - bit 4
7419 00005552 804DFD10          <1>         OR      byte [CMD_BLOCK+5],010H ; SET TO DRIVE 1
7420                            <1>         ;jmp    short drst5
7421 00005556 EBE8              <1>         jmp     short drst3
7422                            <1>
7423                            <1> ;----------------------------------------
7424                            <1> ;       DISK STATUS ROUTINE  (AH = 01H) :
7425                            <1> ;----------------------------------------
7426                            <1>
7427                            <1> RETURN_STATUS:
7428 00005558 A0[8B890100]      <1>         MOV     AL,[DISK_STATUS1]           ; OBTAIN PREVIOUS STATUS
7429 0000555D C605[8B890100]00  <1>         MOV     byte [DISK_STATUS1],0       ; RESET STATUS
7430 00005564 C3                <1>         RETn
7431                            <1>
7432                            <1> ;----------------------------------------
7433                            <1> ;       DISK READ ROUTINE     (AH = 02H) :
7434                            <1> ;----------------------------------------
7435                            <1>
7436                            <1> DISK_READ:
7437 00005565 C645FE20          <1>         MOV     byte [CMD_BLOCK+6],READ_CMD
7438 00005569 E954020000        <1>         JMP     COMMANDI
7439                            <1>
7440                            <1> ;----------------------------------------
7441                            <1> ;       DISK WRITE ROUTINE    (AH = 03H) :
7442                            <1> ;----------------------------------------
7443                            <1>
7444                            <1> DISK_WRITE:
7445 0000556E C645FE30          <1>         MOV     byte [CMD_BLOCK+6],WRITE_CMD
7446 00005572 E9A6020000        <1>         JMP     COMMANDO
7447                            <1>
7448                            <1> ;----------------------------------------
7449                            <1> ;       DISK VERIFY         (AH = 04H) :
7450                            <1> ;----------------------------------------
7451                            <1>
7452                            <1> DISK_VERF:
7453 00005577 C645FE40          <1>         MOV     byte [CMD_BLOCK+6],VERIFY_CMD
7454 0000557B E814030000        <1>         CALL    COMMAND
7455 00005580 750C              <1>         JNZ     short VERF_EXIT             ; CONTROLLER STILL BUSY
7456 00005582 E886030000        <1>         CALL    _WAIT                 ; (Original: CALL WAIT)
7457 00005587 7505              <1>         JNZ     short VERF_EXIT             ; TIME OUT
7458 00005589 E813040000        <1>         CALL    CHECK_STATUS
7459                            <1> VERF_EXIT:
7460 0000558E C3                <1>         RETn
7461                            <1>
7462                            <1> ;----------------------------------------
7463                            <1> ;       FORMATTING          (AH = 05H) :
7464                            <1> ;----------------------------------------
7465                            <1>
7466                            <1> FMT_TRK:                            ; FORMAT TRACK      (AH = 005H)
7467 0000558F C645FE50          <1>         MOV     byte [CMD_BLOCK+6],FMTTRK_CMD
7468                            <1>         ;PUSH   ES
7469                            <1>         ;PUSH   BX
7470 00005593 53                <1>         push    ebx
7471 00005594 E8B7040000        <1>         CALL    GET_VEC             ; GET DISK PARAMETERS ADDRESS
7472                            <1>         ;MOV    AL,[ES:BX+14]       ; GET SECTORS/TRACK
7473 00005599 8A430E            <1>         mov     al, [ebx+14]
7474 0000559C 8845F9            <1>         MOV     [CMD_BLOCK+1],AL    ; SET SECTOR COUNT IN COMMAND
7475 0000559F 5B                <1>         pop     ebx
7476                            <1>         ;POP    BX
7477                            <1>         ;POP    ES
7478 000055A0 E97F020000        <1>         JMP     CMD_OF                  ; GO EXECUTE THE COMMAND
7479                            <1>
7480                            <1> ;----------------------------------------
7481                            <1> ;       READ DASD TYPE          (AH = 15H) :
7482                            <1> ;----------------------------------------
7483                            <1>
7484                            <1> READ_DASD_TYPE:
7485                            <1> READ_D_T:                           ; GET DRIVE PARAMETERS
7486 000055A5 1E                <1>         PUSH    DS                  ; SAVE REGISTERS
7487                            <1>         ;PUSH   ES
7488 000055A6 53                <1>         PUSH    eBX
7489                            <1>         ;CALL   DDS                 ; ESTABLISH ADDRESSING
7490                            <1>         ;push   cs
7491                            <1>         ;pop    ds
7492 000055A7 66BB1000          <1>         mov     bx, KDATA
7493 000055AB 8EDB              <1>         mov     ds, bx
7494                            <1>         ;mov    es, bx
7495 000055AD C605[8B890100]00  <1>         MOV      byte [DISK_STATUS1],0
7496 000055B4 8A1D[8C890100]    <1>         MOV     BL,[HF_NUM]         ; GET NUMBER OF DRIVES
7497 000055BA 80E27F            <1>         AND     DL,7FH              ; GET DRIVE NUMBER
7498 000055BD 38D3              <1>         CMP     BL,DL
7499 000055BF 7627              <1>         JBE     short RDT_NOT_PRESENT     ; RETURN DRIVE NOT PRESENT
7500 000055C1 E88A040000        <1>         CALL    GET_VEC             ; GET DISK PARAMETER ADDRESS
7501                            <1>         ;MOV    AL,[ES:BX+2]        ; HEADS
7502 000055C6 8A4302            <1>         mov     al, [ebx+2]
7503                            <1>         ;MOV    CL,[ES:BX+14]
7504 000055C9 8A4B0E            <1>         mov     cl, [ebx+14]
7505 000055CC F6E9              <1>         IMUL    CL                  ; * NUMBER OF SECTORS
7506                            <1>         ;MOV    CX,[ES:BX]          ; MAX NUMBER OF CYLINDERS
7507 000055CE 668B0B            <1>         mov     cx ,[ebx]
7508                            <1>         ;
7509                            <1>         ; 02/01/2015
7510                            <1>         ; ** leave the last cylinder as reserved for diagnostics **
7511                            <1>         ; (Also in Award BIOS - 1999, AHDSK.ASM, FUN15 -> sub ax, 1)
7512 000055D1 6649              <1>         DEC     CX                  ; LEAVE ONE FOR DIAGNOSTICS
7513                            <1>         ;
7514 000055D3 66F7E9            <1>         IMUL    CX                  ; NUMBER OF SECTORS
```

```
7515 000055D6 6689D1            <1>       MOV    CX,DX               ; HIGH ORDER HALF
7516 000055D9 6689C2            <1>       MOV    DX,AX               ; LOW ORDER HALF
7517                            <1>       ;SUB   AX,AX
7518 000055DC 28C0              <1>       sub    al, al
7519 000055DE B403              <1>       MOV    AH,03H              ; INDICATE FIXED DISK
7520 000055E0 5B                <1> RDT2: POP    eBX                 ; RESTORE REGISTERS
7521                            <1>       ;POP   ES
7522 000055E1 1F                <1>       POP    DS
7523                            <1>       ; (*) CLC                  ; CLEAR CARRY
7524                            <1>       ;RETf  2
7525                            <1>       ; (*) 29/05/2016
7526                            <1>       ; (*) retf 4
7527 000055E2 80642408FE        <1>       and    byte [esp+8], 0FEh ; clear carry bit of eflags register
7528 000055E7 CF                <1>       iretd
7529                            <1>
7530                            <1> RDT_NOT_PRESENT:
7531 000055E8 6629C0            <1>       SUB    AX,AX               ; DRIVE NOT PRESENT RETURN
7532 000055EB 6689C1            <1>       MOV    CX,AX               ; ZERO BLOCK COUNT
7533 000055EE 6689C2            <1>       MOV    DX,AX
7534 000055F1 EBED              <1>       JMP    short RDT2
7535                            <1>
7536                            <1> ; 28/05/2016
7537                            <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
7538                            <1>
7539                            <1> ;----------------------------------------
7540                            <1> ;     GET PARAMETERS           (AH = 08H)
7541                            <1> ;----------------------------------------
7542                            <1>
7543                            <1> GET_PARM_N:
7544                            <1>       ; ebx = user's buffer address for parameters table
7545                            <1> ;GET_PARM:                        ; GET DRIVE PARAMETERS
7546 000055F3 1E                <1>       PUSH   DS                  ; SAVE REGISTERS
7547 000055F4 06                <1>       PUSH   ES
7548 000055F5 53                <1>       PUSH   eBX
7549                            <1>       ;MOV   AX,ABS0             ; ESTABLISH ADDRESSING
7550                            <1>       ;MOV   DS,AX
7551                            <1>       ;TEST  DL,1                ; CHECK FOR DRIVE 1
7552                            <1>       ;JZ    short G0
7553                            <1>       ;LES   BX,@HF1_TBL_VEC
7554                            <1>       ;JMP   SHORT G1
7555                            <1> ;G0:  LES    BX,@HF_TBL_VEC
7556                            <1> ;G1:
7557                            <1>       ;CALL  DDS                 ; ESTABLISH SEGMENT
7558                            <1>       ; 22/12/2014
7559                            <1>       ;push  cs
7560                            <1>       ;pop   ds
7561 000055F6 66BB1000         <1>       mov    bx, KDATA
7562 000055FA 8EDB              <1>       mov    ds, bx
7563 000055FC 8EC3              <1>       mov    es, bx ; 27/05/2016
7564                            <1>       ;
7565 000055FE 80EA80            <1>       SUB    DL,80H
7566 00005601 80FA04            <1>       CMP    DL,MAX_FILE         ; TEST WITHIN RANGE
7567 00005604 7361              <1>       JAE    short G4
7568                            <1>       ;
7569 00005606 31DB              <1>       xor    ebx, ebx ; 21/02/2015
7570                            <1>       ; 22/12/2014
7571 00005608 88D3              <1>       mov    bl, dl
7572                            <1>       ;xor   bh, bh
7573 0000560A C0E302            <1>       shl    bl, 2               ; convert index to offset
7574                            <1>       ;add   bx, HF_TBL_VEC
7575 0000560D 81C3[90890100]    <1>       add    ebx, HF_TBL_VEC
7576                            <1>       ;mov   ax, [bx+2]
7577                            <1>       ;mov   es, ax              ; dpt segment
7578                            <1>       ;mov   bx, [bx]            ; dpt offset
7579 00005613 8B1B              <1>       mov    ebx, [ebx] ; 32 bit offset
7580                            <1>
7581 00005615 C605[8B890100]00  <1>       MOV    byte [DISK_STATUS1],0
7582                            <1>       ;MOV  AX,[ES:BX]           ; MAX NUMBER OF CYLINDERS
7583 0000561C 668B03            <1>       mov    ax, [ebx]
7584                            <1>       ;;SUB  AX,2                ; ADJUST FOR 0-N
7585 0000561F 6648              <1>       dec    ax                  ; max. cylinder number
7586 00005621 88C5              <1>       MOV    CH,AL
7587 00005623 66250003          <1>       AND    AX,0300H            ; HIGH TWO BITS OF CYLINDER
7588 00005627 66D1E8            <1>       SHR    AX,1
7589 0000562A 66D1E8            <1>       SHR    AX,1
7590                            <1>       ;OR    AL,[ES:BX+14]       ; SECTORS
7591 0000562D 0A430E            <1>       or     al, [ebx+14]
7592 00005630 88C1              <1>       MOV    CL,AL
7593                            <1>       ;MOV   DH,[ES:BX+2]        ; HEADS
7594 00005632 8A7302            <1>       mov    dh, [ebx+2]
7595 00005635 FECE              <1>       DEC    DH                  ; 0-N RANGE
7596 00005637 8A15[8C890100]    <1>       MOV    DL,[HF_NUM]         ; DRIVE COUNT
7597 0000563D 6629C0            <1>       SUB    AX,AX
7598                            <1>         ;27/12/2014
7599                            <1>       ;mov   di, bx              ; HDPT offset
7600                            <1>
7601                            <1>       ; 27/05/2016
7602                            <1>       ; return fixed disk parameters table to user
7603                            <1>       ; in user's buffer, which is pointed by EBX
7604                            <1>       ;
7605 00005640 873C24            <1>       xchg   edi, [esp]          ; ebx (input)-> edi, edi -> [esp]
7606 00005643 56                <1>       push   esi
7607 00005644 89DE              <1>       mov    esi, ebx            ; hard disk parameter table (32 bytes)
7608 00005646 89FB              <1>       mov    ebx, edi            ; ebx = user's buffer address
7609 00005648 51                <1>       push   ecx
7610 00005649 50                <1>       push   eax
7611 0000564A B920000000        <1>       mov    ecx, 32 ; 32 bytes
7612 0000564F E8D0C30000        <1>       call   transfer_to_user_buffer ; trdosk6.s (16/05/2016)
7613 00005654 58                <1>       pop    eax
7614 00005655 59                <1>       pop    ecx
7615 00005656 5E                <1>       pop    esi
7616 00005657 5F                <1>       pop    edi
7617 00005658 730A              <1>       jnc    short G5
7618                            <1>       ; 29/05/2016 (*)
7619 0000565A B8FF000000        <1>       mov    eax, 0FFh ; unknown error !
```

```
7620                                    <1> _G6:
7621 0000565F 804C241001                <1>         or    byte [esp+16], 1 ; set carry bit of eflags register
7622                                    <1> G5:
7623                                    <1>         ; 27/05/2016
7624                                    <1>         ;POP   eBX                ; RESTORE REGISTERS
7625 00005664 07                        <1>         POP    ES
7626 00005665 1F                        <1>         POP    DS
7627                                    <1>         ;RETf  2
7628                                    <1>         ; (*) 29/05/2016
7629                                    <1>         ; (*) retf 4
7630                                    <1>         ; (*) or byte [esp+8], 1 ; set carry bit of eflags register
7631 00005666 CF                        <1>         iretd
7632                                    <1> G4:
7633 00005667 C605[8B890100]07          <1>         MOV     byte [DISK_STATUS1],INIT_FAIL ; OPERATION FAILED
7634 0000566E B407                      <1>         MOV    AH,INIT_FAIL
7635 00005670 28C0                      <1>         SUB    AL,AL
7636 00005672 6629D2                    <1>         SUB    DX,DX
7637 00005675 6629C9                    <1>         SUB    CX,CX
7638                                    <1>         ; 29/05/2016 (*)
7639                                    <1>         ;STC                      ; SET ERROR FLAG
7640                                    <1>         ;JMP   short G5
7641 00005678 EBE5                      <1>         jmp    short _G6
7642                                    <1>
7643                                    <1> ;----------------------------------------
7644                                    <1> ;     INITIALIZE DRIVE     (AH = 09H) :
7645                                    <1> ;----------------------------------------
7646                                    <1>         ; 03/01/2015
7647                                    <1>         ; According to ATA-ATAPI specification v2.0 to v5.0
7648                                    <1>         ; logical sector per logical track
7649                                    <1>         ; and logical heads - 1 would be set but
7650                                    <1>         ; it is seen as it will be good
7651                                    <1>         ; if physical parameters will be set here
7652                                    <1>         ; because, number of heads <= 16.
7653                                    <1>         ; (logical heads usually more than 16)
7654                                    <1>         ; NOTE: ATA logical parameters (software C, H, S)
7655                                    <1>         ;      == INT 13h physical parameters
7656                                    <1>
7657                                    <1> ;INIT_DRV:
7658                                    <1> ;     MOV    byte [CMD_BLOCK+6],SET_PARM_CMD
7659                                    <1> ;     CALL   GET_VEC             ; ES:BX -> PARAMETER BLOCK
7660                                    <1> ;     MOV    AL,[ES:BX+2]        ; GET NUMBER OF HEADS
7661                                    <1> ;     DEC    AL                  ; CONVERT TO 0-INDEX
7662                                    <1> ;     MOV    AH,[CMD_BLOCK+5]    ; GET SDH REGISTER
7663                                    <1> ;     AND    AH,0F0H             ; CHANGE HEAD NUMBER
7664                                    <1> ;     OR     AH,AL               ; TO MAX HEAD
7665                                    <1> ;     MOV    [CMD_BLOCK+5],AH
7666                                    <1> ;     MOV    AL,[ES:BX+14]       ; MAX SECTOR NUMBER
7667                                    <1> ;     MOV    [CMD_BLOCK+1],AL
7668                                    <1> ;     SUB    AX,AX
7669                                    <1> ;     MOV    [CMD_BLOCK+3],AL    ; ZERO FLAGS
7670                                    <1> ;     CALL   COMMAND             ; TELL CONTROLLER
7671                                    <1> ;     JNZ    short INIT_EXIT           ; CONTROLLER BUSY ERROR
7672                                    <1> ;     CALL   NOT_BUSY            ; WAIT FOR IT TO BE DONE
7673                                    <1> ;     JNZ    short INIT_EXIT           ; TIME OUT
7674                                    <1> ;     CALL   CHECK_STATUS
7675                                    <1> ;INIT_EXIT:
7676                                    <1> ;     RETn
7677                                    <1>
7678                                    <1> ; 04/01/2015
7679                                    <1> ; 02/01/2015 - Derived from from AWARD BIOS 1999
7680                                    <1> ;                          AHDSK.ASM - INIT_DRIVE
7681                                    <1> INIT_DRV:
7682                                    <1>         ;xor   ah,ah
7683 0000567A 31C0                      <1>         xor    eax, eax ; 21/02/2015
7684 0000567C B00B                      <1>         mov    al,11 ; Physical heads from translated HDPT
7685 0000567E 3825[A0890100]            <1>         cmp    [LBAMode], ah   ; 0
7686 00005684 7702                      <1>         ja     short idrv0
7687 00005686 B002                      <1>         mov    al,2  ; Physical heads from standard HDPT
7688                                    <1> idrv0:
7689                                    <1>         ; DL = drive number (0 based)
7690 00005688 E8C3030000                <1>         call   GET_VEC
7691                                    <1>         ;push  bx
7692 0000568D 53                        <1>         push   ebx ; 21/02/2015
7693                                    <1>         ;add   bx,ax
7694 0000568E 01C3                      <1>         add    ebx, eax
7695                                    <1>         ;; 05/01/2015
7696 00005690 8A25[C46C0000]            <1>         mov    ah, [hf_m_s] ; drive number (0= master, 1= slave)
7697                                    <1>         ;;and  ah,1
7698 00005696 C0E404                    <1>         shl    ah,4
7699 00005699 80CCA0                    <1>         or     ah,0A0h  ; Drive/Head register - 10100000b (A0h)
7700                                    <1>         ;mov   al,[es:bx]
7701 0000569C 8A03                      <1>         mov    al, [ebx] ; 21/02/2015
7702 0000569E FEC8                      <1>         dec    al       ; last head number
7703                                    <1>         ;and   al,0Fh
7704 000056A0 08E0                      <1>         or     al,ah   ; lower 4 bits for head number
7705                                    <1>         ;
7706 000056A2 C645FE91                  <1>         mov    byte [CMD_BLOCK+6],SET_PARM_CMD
7707 000056A6 8845FD                    <1>         mov    [CMD_BLOCK+5],al
7708                                    <1>         ;pop   bx
7709 000056A9 5B                        <1>         pop    ebx
7710 000056AA 29C0                      <1>         sub    eax, eax ; 21/02/2015
7711 000056AC B004                      <1>         mov    al,4 ; Physical sec per track from translated HDPT
7712 000056AE 803D[A0890100]00          <1>         cmp    byte [LBAMode], 0
7713 000056B5 7702                      <1>         ja     short idrv1
7714 000056B7 B00E                      <1>         mov    al,14 ; Physical sec per track from standard HDPT
7715                                    <1> idrv1:
7716                                    <1>         ;xor   ah,ah
7717                                    <1>         ;add   bx,ax
7718 000056B9 01C3                      <1>         add    ebx, eax ; 21/02/2015
7719                                    <1>         ;mov   al,[es:bx]
7720                                    <1>                ; sector number
7721 000056BB 8A03                      <1>         mov    al, [ebx]
7722 000056BD 8845F9                    <1>         mov    [CMD_BLOCK+1],al
7723 000056C0 28C0                      <1>         sub    al,al
7724 000056C2 8845FB                    <1>         mov    [CMD_BLOCK+3],al  ; ZERO FLAGS
```

```
7725 000056C5 E8CA010000          <1>       call   COMMAND        ; TELL CONTROLLER
7726 000056CA 750C                <1>       jnz    short INIT_EXIT       ; CONTROLLER BUSY ERROR
7727 000056CC E878020000          <1>       call   NOT_BUSY       ; WAIT FOR IT TO BE DONE
7728 000056D1 7505                <1>       jnz    short INIT_EXIT       ; TIME OUT
7729 000056D3 E8C9020000          <1>       call   CHECK_STATUS
7730                              <1> INIT_EXIT:
7731 000056D8 C3                  <1>       RETn
7732                              <1>
7733                              <1> ;----------------------------------------
7734                              <1> ;     READ LONG         (AH = 0AH) :
7735                              <1> ;----------------------------------------
7736                              <1>
7737                              <1> RD_LONG:
7738                              <1>       ;MOV   @CMD_BLOCK+6,READ_CMD OR ECC_MODE
7739 000056D9 C645FE22            <1>       mov    byte [CMD_BLOCK+6],READ_CMD + ECC_MODE
7740 000056DD E9E0000000          <1>       JMP    COMMANDI
7741                              <1>
7742                              <1> ;----------------------------------------
7743                              <1> ;     WRITE LONG        (AH = 0BH) :
7744                              <1> ;----------------------------------------
7745                              <1>
7746                              <1> WR_LONG:
7747                              <1>       ;MOV   @CMD_BLOCK+6,WRITE_CMD OR ECC_MODE
7748 000056E2 C645FE32            <1>       MOV    byte [CMD_BLOCK+6],WRITE_CMD + ECC_MODE
7749 000056E6 E932010000          <1>       JMP    COMMANDO
7750                              <1>
7751                              <1> ;----------------------------------------
7752                              <1> ;     SEEK              (AH = 0CH) :
7753                              <1> ;----------------------------------------
7754                              <1>
7755                              <1> DISK_SEEK:
7756 000056EB C645FE70            <1>       MOV    byte [CMD_BLOCK+6],SEEK_CMD
7757 000056EF E8A0010000          <1>       CALL   COMMAND
7758 000056F4 751C                <1>       JNZ    short DS_EXIT       ; CONTROLLER BUSY ERROR
7759 000056F6 E812020000          <1>       CALL   WAIT
7760 000056FB 7515                <1>       JNZ    DS_EXIT             ; TIME OUT ON SEEK
7761 000056FD E89F020000          <1>       CALL   CHECK_STATUS
7762 00005702 803D[8B890100]40    <1>       CMP    byte [DISK_STATUS1],BAD_SEEK
7763 00005709 7507                <1>       JNE    short DS_EXIT
7764 0000570B C605[8B890100]00    <1>       MOV    byte [DISK_STATUS1],0
7765                              <1> DS_EXIT:
7766 00005712 C3                  <1>       RETn
7767                              <1>
7768                              <1> ;----------------------------------------
7769                              <1> ;     TEST DISK READY      (AH = 10H) :
7770                              <1> ;----------------------------------------
7771                              <1>
7772                              <1> TST_RDY:                          ; WAIT FOR CONTROLLER
7773 00005713 E831020000          <1>       CALL   NOT_BUSY
7774 00005718 751C                <1>       JNZ    short TR_EX
7775 0000571A 8A45FD              <1>       MOV    AL,[CMD_BLOCK+5]    ; SELECT DRIVE
7776 0000571D 668B15[C06C0000]    <1>       MOV    DX,[HF_PORT]
7777 00005724 80C206              <1>       add    dl,6
7778 00005727 EE                  <1>       OUT    DX,AL
7779 00005728 E88C020000          <1>       CALL   CHECK_ST            ; CHECK STATUS ONLY
7780 0000572D 7507                <1>       JNZ    short TR_EX
7781 0000572F C605[8B890100]00    <1>       MOV    byte [DISK_STATUS1],0 ; WIPE OUT DATA CORRECTED ERROR
7782                              <1> TR_EX:
7783 00005736 C3                  <1>       RETn
7784                              <1>
7785                              <1> ;----------------------------------------
7786                              <1> ;     RECALIBRATE       (AH = 11H) :
7787                              <1> ;----------------------------------------
7788                              <1>
7789                              <1> HDISK_RECAL:
7790 00005737 C645FE10            <1>       MOV   byte [CMD_BLOCK+6],RECAL_CMD ; 10h, 16
7791 0000573B E854010000          <1>       CALL   COMMAND             ; START THE OPERATION
7792 00005740 7523                <1>       JNZ    short RECAL_EXIT    ; ERROR
7793 00005742 E8C6010000          <1>       CALL   _WAIT               ; WAIT FOR COMPLETION
7794 00005747 7407                <1>       JZ     short RECAL_X             ; TIME OUT ONE OK ?
7795 00005749 E8BF010000          <1>       CALL   _WAIT               ; WAIT FOR COMPLETION LONGER
7796 0000574E 7515                <1>       JNZ    short RECAL_EXIT    ; TIME OUT TWO TIMES IS ERROR
7797                              <1> RECAL_X:
7798 00005750 E84C020000          <1>       CALL   CHECK_STATUS
7799 00005755 803D[8B890100]40    <1>       CMP    byte [DISK_STATUS1],BAD_SEEK ; SEEK NOT COMPLETE
7800 0000575C 7507                <1>       JNE    short RECAL_EXIT    ; IS OK
7801 0000575E C605[8B890100]00    <1>       MOV    byte [DISK_STATUS1],0
7802                              <1> RECAL_EXIT:
7803 00005765 803D[8B890100]00    <1>       CMP   byte [DISK_STATUS1],0
7804 0000576C C3                  <1>       RETn
7805                              <1>
7806                              <1> ;----------------------------------------
7807                              <1> ;      CONTROLLER DIAGNOSTIC (AH = 14H) :
7808                              <1> ;----------------------------------------
7809                              <1>
7810                              <1> CTLR_DIAGNOSTIC:
7811 0000576D FA                  <1>       CLI                         ; DISABLE INTERRUPTS WHILE CHANGING MASK
7812 0000576E E4A1                <1>       IN     AL,INTB01            ; TURN ON SECOND INTERRUPT CHIP
7813                              <1>       ;AND   AL,0BFH
7814 00005770 243F                <1>       and    al, 3Fh              ; enable IRQ 14 & IRQ 15
7815                              <1>       ;JMP   $+2
7816                              <1>       IODELAY
2968 00005772 EB00                <2>  jmp short $+2
2969 00005774 EB00                <2>  jmp short $+2
7817 00005776 E6A1                <1>       OUT    INTB01,AL
7818                              <1>       IODELAY
2968 00005778 EB00                <2>  jmp short $+2
2969 0000577A EB00                <2>  jmp short $+2
7819 0000577C E421                <1>       IN     AL,INTA01            ; LET INTERRUPTS PASS THRU TO
7820 0000577E 24FB                <1>       AND    AL,0FBH              ;  SECOND CHIP
7821                              <1>       ;JMP   $+2
7822                              <1>       IODELAY
2968 00005780 EB00                <2>  jmp short $+2
2969 00005782 EB00                <2>  jmp short $+2
7823 00005784 E621                <1>       OUT    INTA01,AL
```

```
7824 00005786 FB                  <1>       STI
7825 00005787 E8BD010000          <1>       CALL   NOT_BUSY           ; WAIT FOR CARD
7826 0000578C 752B                <1>       JNZ    short CD_ERR       ; BAD CARD
7827                              <1>       ;MOV   DX, HF_PORT+7
7828 0000578E 668B15[C06C0000]    <1>       mov    dx, [HF_PORT]
7829 00005795 80C207              <1>       add    dl, 7
7830 00005798 B090                <1>       MOV    AL,DIAG_CMD        ; START DIAGNOSE
7831 0000579A EE                  <1>       OUT    DX,AL
7832 0000579B E8A9010000          <1>       CALL   NOT_BUSY           ; WAIT FOR IT TO COMPLETE
7833 000057A0 B480                <1>       MOV    AH,TIME_OUT
7834 000057A2 7517                <1>       JNZ    short CD_EXIT            ; TIME OUT ON DIAGNOSTIC
7835                              <1>       ;MOV   DX,HF_PORT+1       ; GET ERROR REGISTER
7836 000057A4 668B15[C06C0000]    <1>       mov    dx, [HF_PORT]
7837 000057AB FEC2                <1>       inc    dl
7838 000057AD EC                  <1>       IN     AL,DX
7839 000057AE A2[82890100]        <1>       MOV    [HF_ERROR],AL      ; SAVE IT
7840 000057B3 B400                <1>       MOV    AH,0
7841 000057B5 3C01                <1>       CMP    AL,1               ; CHECK FOR ALL OK
7842 000057B7 7402                <1>       JE     SHORT CD_EXIT
7843 000057B9 B420                <1> CD_ERR: MOV  AH,BAD_CNTLR
7844                              <1> CD_EXIT:
7845 000057BB 8825[8B890100]      <1>       MOV    [DISK_STATUS1],AH
7846 000057C1 C3                  <1>       RETn
7847                              <1>
7848                              <1> ;---------------------------------------
7849                              <1> ; COMMANDI                           :
7850                              <1> ;     REPEATEDLY INPUTS DATA TILL    :
7851                              <1> ;     NSECTOR RETURNS ZERO           :
7852                              <1> ;---------------------------------------
7853                              <1> COMMANDI:
7854 000057C2 E862020000          <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
7855 000057C7 7253                <1>       JC     short CMD_ABORT
7856                              <1>       ;MOV   DI,BX
7857 000057C9 89DF                <1>       mov    edi, ebx ; 21/02/2015
7858 000057CB E8C4000000          <1>       CALL   COMMAND            ; OUTPUT COMMAND
7859 000057D0 754A                <1>       JNZ    short CMD_ABORT
7860                              <1> CMD_I1:
7861 000057D2 E836010000          <1>       CALL   _WAIT              ; WAIT FOR DATA REQUEST INTERRUPT
7862 000057D7 7543                <1>       JNZ    short TM_OUT       ; TIME OUT
7863                              <1> cmd_i1x: ; 18/02/2016
7864                              <1>       ;MOV   CX,256             ; SECTOR SIZE IN WORDS
7865 000057D9 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
7866                              <1>       ;MOV   DX,HF_PORT
7867 000057DE 668B15[C06C0000]    <1>       mov    dx,[HF_PORT]
7868 000057E5 FA                  <1>       CLI
7869 000057E6 FC                  <1>       CLD
7870 000057E7 F3666D              <1>       REP    INSW               ; GET THE SECTOR
7871 000057EA FB                  <1>       STI
7872 000057EB F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL INPUT
7873 000057EF 7419                <1>       JZ     short CMD_I3
7874 000057F1 E880010000          <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
7875 000057F6 7224                <1>       JC     short TM_OUT
7876                              <1>       ;MOV   DX,HF_PORT
7877 000057F8 668B15[C06C0000]    <1>       mov    dx,[HF_PORT]
7878                              <1>       ;MOV   CX,4               ; GET ECC BYTES
7879 000057FF B904000000          <1>       mov    ecx, 4 ; mov cx, 4
7880 00005804 EC                  <1> CMD_I2: IN   AL,DX
7881                              <1>       ;MOV   [ES:DI],AL         ; GO SLOW FOR BOARD
7882 00005805 8807                <1>       mov    [edi], al ; 21/02/2015
7883 00005807 47                  <1>       INC    eDI
7884 00005808 E2FA                <1>       LOOP   CMD_I2
7885                              <1> CMD_I3:
7886                              <1>       ; wait for 400 ns
7887 0000580A 80C207              <1>       add    dl, 7
7888 0000580D EC                  <1>       in     al, dx
7889 0000580E EC                  <1>       in     al, dx
7890 0000580F EC                  <1>       in     al, dx
7891                              <1>       ;
7892 00005810 E88C010000          <1>       CALL   CHECK_STATUS
7893 00005815 7505                <1>       JNZ    short CMD_ABORT          ; ERROR RETURNED
7894 00005817 FE4DF9              <1>       DEC    byte [CMD_BLOCK+1]  ; CHECK FOR MORE
7895                              <1>       ;JNZ   SHORT CMD_I1
7896 0000581A 75BD                <1>       jnz    short cmd_i1x ; 18/02/2016
7897                              <1> CMD_ABORT:
7898 0000581C C3                  <1> TM_OUT: RETn
7899                              <1>
7900                              <1> ;---------------------------------------
7901                              <1> ; COMMANDO                           :
7902                              <1> ;     REPEATEDLY OUTPUTS DATA TILL   :
7903                              <1> ;     NSECTOR RETURNS ZERO           :
7904                              <1> ;---------------------------------------
7905                              <1> COMMANDO:
7906 0000581D E807020000          <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
7907 00005822 72F8                <1>       JC     short CMD_ABORT
7908 00005824 89DE                <1> CMD_OF: MOV  eSI,eBX ; 21/02/2015
7909 00005826 E869000000          <1>       CALL   COMMAND            ; OUTPUT COMMAND
7910 0000582B 75EF                <1>       JNZ    short CMD_ABORT
7911 0000582D E844010000          <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
7912 00005832 72E8                <1>       JC     short TM_OUT             ; TOO LONG
7913                              <1> CMD_O1: ;PUSH      DS
7914                              <1>       ;PUSH  ES                 ; MOVE ES TO DS
7915                              <1>       ;POP   DS
7916                              <1>       ;MOV   CX,256             ; PUT THE DATA OUT TO THE CARD
7917                              <1>       ;MOV   DX,HF_PORT
7918                              <1>       ; 01/02/2015
7919 00005834 668B15[C06C0000]    <1>       mov    dx, [HF_PORT]
7920                              <1>       ;push  es
7921                              <1>       ;pop   ds
7922                              <1>       ;mov   cx, 256
7923 0000583B B900010000          <1>       mov    ecx, 256 ; 21/02/2015
7924 00005840 FA                  <1>       CLI
7925 00005841 FC                  <1>       CLD
7926 00005842 F3666F              <1>       REP    OUTSW
7927 00005845 FB                  <1>       STI
7928                              <1>       ;POP   DS                 ; RESTORE DS
```

```
7929 00005846 F645FE02           <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL OUTPUT
7930 0000584A 7419               <1>       JZ     short CMD_O3
7931 0000584C E825010000         <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
7932 00005851 72C9               <1>       JC     short TM_OUT
7933                             <1>       ;MOV   DX,HF_PORT
7934 00005853 668B15[C06C0000]   <1>       mov    dx, [HF_PORT]
7935                             <1>       ;MOV   CX,4               ; OUTPUT THE ECC BYTES
7936 0000585A B904000000         <1>       mov    ecx, 4  ; mov cx, 4
7937                             <1> CMD_O2: ;MOV AL,[ES:SI]
7938 0000585F 8A06               <1>       mov    al, [esi]
7939 00005861 EE                 <1>       OUT    DX,AL
7940 00005862 46                 <1>       INC    eSI
7941 00005863 E2FA               <1>       LOOP   CMD_O2
7942                             <1> CMD_O3:
7943 00005865 E8A3000000         <1>       CALL   _WAIT              ; WAIT FOR SECTOR COMPLETE INTERRUPT
7944 0000586A 75B0               <1>       JNZ    short TM_OUT       ; ERROR RETURNED
7945 0000586C E830010000         <1>       CALL   CHECK_STATUS
7946 00005871 75A9               <1>       JNZ    short CMD_ABORT
7947 00005873 F605[81890100]08   <1>       TEST   byte [HF_STATUS],ST_DRQ   ; CHECK FOR MORE
7948 0000587A 75B8               <1>       JNZ    SHORT CMD_O1
7949                             <1>       ;MOV   DX,HF_PORT+2       ; CHECK RESIDUAL SECTOR COUNT
7950 0000587C 668B15[C06C0000]   <1>       mov    dx, [HF_PORT]
7951                             <1>       ;add   dl, 2
7952 00005883 FEC2               <1>       inc    dl
7953 00005885 FEC2               <1>       inc    dl
7954 00005887 EC                 <1>       IN     AL,DX              ;
7955 00005888 A8FF               <1>       TEST   AL,0FFH            ;
7956 0000588A 7407               <1>       JZ     short CMD_O4              ; COUNT = 0  OK
7957 0000588C C605[8B890100]BB   <1>       MOV    byte [DISK_STATUS1],UNDEF_ERR
7958                             <1>                                 ; OPERATION ABORTED - PARTIAL TRANSFER
7959                             <1> CMD_O4:
7960 00005893 C3                 <1>       RETn
7961                             <1>
7962                             <1> ;--------------------------------------------------------
7963                             <1> ; COMMAND                                                :
7964                             <1> ;     THIS ROUTINE OUTPUTS THE COMMAND BLOCK             :
7965                             <1> ; OUTPUT                                                 :
7966                             <1> ;     BL = STATUS                                        :
7967                             <1> ;     BH = ERROR REGISTER                                :
7968                             <1> ;--------------------------------------------------------
7969                             <1>
7970                             <1> COMMAND:
7971 00005894 53                 <1>       PUSH   eBX                ; WAIT FOR SEEK COMPLETE AND READY
7972                             <1>       ;;MOV  CX,DELAY_2         ; SET INITIAL DELAY BEFORE TEST
7973                             <1> COMMAND1:
7974                             <1>       ;;PUSH CX                 ; SAVE LOOP COUNT
7975 00005895 E879FEFFFF         <1>       CALL   TST_RDY            ; CHECK DRIVE READY
7976                             <1>       ;;POP  CX
7977 0000589A 7419               <1>       JZ     short COMMAND2             ; DRIVE IS READY
7978 0000589C 803D[8B890100]80   <1>         CMP     byte [DISK_STATUS1],TIME_OUT ; TST_RDY TIMED OUT--GIVE UP
7979                             <1>       ;JZ    short CMD_TIMEOUT
7980                             <1>       ;;LOOP COMMAND1           ; KEEP TRYING FOR A WHILE
7981                             <1>       ;JMP   SHORT COMMAND4            ; ITS NOT GOING TO GET READY
7982 000058A3 7507               <1>       jne    short COMMAND4
7983                             <1> CMD_TIMEOUT:
7984 000058A5 C605[8B890100]20   <1>       MOV    byte [DISK_STATUS1],BAD_CNTLR
7985                             <1> COMMAND4:
7986 000058AC 5B                 <1>       POP    eBX
7987 000058AD 803D[8B890100]00   <1>         CMP     byte [DISK_STATUS1],0   ; SET CONDITION CODE FOR CALLER
7988 000058B4 C3                 <1>       RETn
7989                             <1> COMMAND2:
7990 000058B5 5B                 <1>       POP    eBX
7991 000058B6 57                 <1>       PUSH   eDI
7992 000058B7 C605[83890100]00   <1>       MOV    byte [HF_INT_FLAG],0      ; RESET INTERRUPT FLAG
7993 000058BE FA                 <1>       CLI                       ; INHIBIT INTERRUPTS WHILE CHANGING MASK
7994 000058BF E4A1               <1>       IN     AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
7995                             <1>       ;AND   AL,0BFH
7996 000058C1 243F               <1>       and    al, 3Fh                   ; Enable IRQ 14 & 15
7997                             <1>       ;JMP   $+2
7998                             <1>       IODELAY
2968 000058C3 EB00               <2>  jmp short $+2
2969 000058C5 EB00               <2>  jmp short $+2
7999 000058C7 E6A1               <1>       OUT    INTB01,AL
8000 000058C9 E421               <1>       IN     AL,INTA01          ; LET INTERRUPTS PASS THRU TO
8001 000058CB 24FB               <1>       AND    AL,0FBH            ;  SECOND CHIP
8002                             <1>       ;JMP   $+2
8003                             <1>       IODELAY
2968 000058CD EB00               <2>  jmp short $+2
2969 000058CF EB00               <2>  jmp short $+2
8004 000058D1 E621               <1>       OUT    INTA01,AL
8005 000058D3 FB                 <1>       STI
8006 000058D4 31FF               <1>       XOR    eDI,eDI                   ; INDEX THE COMMAND TABLE
8007                             <1>       ;MOV   DX,HF_PORT+1       ; DISK ADDRESS
8008 000058D6 668B15[C06C0000]   <1>       mov    dx, [HF_PORT]
8009 000058DD FEC2               <1>       inc    dl
8010 000058DF F605[8D890100]C0   <1>       TEST   byte [CONTROL_BYTE],0C0H ; CHECK FOR RETRY SUPPRESSION
8011 000058E6 7411               <1>       JZ     short COMMAND3
8012 000058E8 8A45FE             <1>       MOV    AL, [CMD_BLOCK+6]   ; YES-GET OPERATION CODE
8013 000058EB 24F0               <1>       AND    AL,0F0H             ; GET RID OF MODIFIERS
8014 000058ED 3C20               <1>       CMP    AL,20H              ; 20H-40H IS READ, WRITE, VERIFY
8015 000058EF 7208               <1>       JB     short COMMAND3
8016 000058F1 3C40               <1>       CMP    AL,40H
8017 000058F3 7704               <1>       JA     short COMMAND3
8018 000058F5 804DFE01           <1>       OR     byte [CMD_BLOCK+6],NO_RETRIES
8019                             <1>                                 ; VALID OPERATION FOR RETRY SUPPRESS
8020                             <1> COMMAND3:
8021 000058F9 8A443DF8           <1>       MOV    AL,[CMD_BLOCK+eDI]  ; GET THE COMMAND STRING BYTE
8022 000058FD EE                 <1>       OUT    DX,AL               ; GIVE IT TO CONTROLLER
8023                             <1>       IODELAY
2968 000058FE EB00               <2>  jmp short $+2
2969 00005900 EB00               <2>  jmp short $+2
8024 00005902 47                 <1>       INC    eDI                 ; NEXT BYTE IN COMMAND BLOCK
8025 00005903 6642               <1>       INC    DX                  ; NEXT DISK ADAPTER REGISTER
8026 00005905 6683FF07           <1>       cmp    di, 7  ; 1/1/2015   ; ALL DONE?
8027 00005909 75EE               <1>       JNZ    short COMMAND3            ; NO--GO DO NEXT ONE
```

```
8028 0000590B 5F                <1>       POP    eDI
8029 0000590C C3                <1>       RETn                        ; ZERO FLAG IS SET
8030                            <1>
8031                            <1> ;CMD_TIMEOUT:
8032                            <1> ;     MOV    byte [DISK_STATUS1],BAD_CNTLR
8033                            <1> ;COMMAND4:
8034                            <1> ;     POP    BX
8035                            <1> ;     CMP    [DISK_STATUS1],0    ; SET CONDITION CODE FOR CALLER
8036                            <1> ;     RETn
8037                            <1>
8038                            <1> ;----------------------------------------
8039                            <1> ;     WAIT FOR INTERRUPT        :
8040                            <1> ;----------------------------------------
8041                            <1> ;WAIT:
8042                            <1> _WAIT:
8043 0000590D FB                <1>       STI                         ; MAKE SURE INTERRUPTS ARE ON
8044                            <1>       ;SUB   CX,CX                ; SET INITIAL DELAY BEFORE TEST
8045                            <1>       ;CLC
8046                            <1>       ;MOV   AX,9000H             ; DEVICE WAIT INTERRUPT
8047                            <1>       ;INT   15H
8048                            <1>       ;JC    WT2                  ; DEVICE TIMED OUT
8049                            <1>       ;MOV   BL,DELAY_1           ; SET DELAY COUNT
8050                            <1>
8051                            <1>       ;mov   bl, WAIT_HDU_INT_HI
8052                            <1>       ;; 21/02/2015
8053                            <1>       ;;mov  bl, WAIT_HDU_INT_HI + 1
8054                            <1>       ;;mov  cx, WAIT_HDU_INT_LO
8055 0000590E B915160500        <1>       mov    ecx, WAIT_HDU_INT_LH
8056                            <1>                                   ; (AWARD BIOS -> WAIT_FOR_MEM)
8057                            <1> ;-----      WAIT LOOP
8058                            <1>
8059                            <1> WT1:
8060                            <1>       ;TEST  byte [HF_INT_FLAG],80H     ; TEST FOR INTERRUPT
8061 00005913 F605[83890100]C0  <1>       test   byte [HF_INT_FLAG],0C0h
8062                            <1>       ;LOOPZ WT1
8063 0000591A 7517              <1>       JNZ    short WT3            ; INTERRUPT--LETS GO
8064                            <1>       ;DEC   BL
8065                            <1>       ;JNZ   short WT1            ; KEEP TRYING FOR A WHILE
8066                            <1>
8067                            <1> WT1_hi:
8068 0000591C E461              <1>       in     al, SYS1 ; 61h (PORT_B)    ; wait for lo to hi
8069 0000591E A810              <1>       test   al, 10h                    ; transition on memory
8070 00005920 75FA              <1>       jnz    short WT1_hi         ; refresh.
8071                            <1> WT1_lo:
8072 00005922 E461              <1>       in     al, SYS1             ; 061h (PORT_B)
8073 00005924 A810              <1>       test   al, 10h
8074 00005926 74FA              <1>       jz     short WT1_lo
8075 00005928 E2E9              <1>       loop   WT1
8076                            <1>       ;;or   bl, bl
8077                            <1>       ;;jz   short WT2
8078                            <1>       ;;dec  bl
8079                            <1>       ;;jmp  short WT1
8080                            <1>       ;dec   bl
8081                            <1>       ;jnz   short WT1
8082                            <1>
8083 0000592A C605[8B890100]80  <1> WT2:  MOV    byte [DISK_STATUS1],TIME_OUT ; REPORT TIME OUT ERROR
8084 00005931 EB0E              <1>       JMP    SHORT WT4
8085 00005933 C605[8B890100]00  <1> WT3:  MOV    byte [DISK_STATUS1],0
8086 0000593A C605[83890100]00  <1>       MOV    byte [HF_INT_FLAG],0
8087 00005941 803D[8B890100]00  <1> WT4:  CMP    byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
8088 00005948 C3                <1>       RETn
8089                            <1>
8090                            <1> ;----------------------------------------
8091                            <1> ;     WAIT FOR CONTROLLER NOT BUSY     :
8092                            <1> ;----------------------------------------
8093                            <1> NOT_BUSY:
8094 00005949 FB                <1>       STI                         ; MAKE SURE INTERRUPTS ARE ON
8095                            <1>       ;PUSH  eBX
8096                            <1>       ;SUB   CX,CX                ; SET INITIAL DELAY BEFORE TEST
8097 0000594A 668B15[C06C0000]  <1>       mov    DX, [HF_PORT]
8098 00005951 80C207            <1>       add    dl, 7                ; Status port (HF_PORT+7)
8099                            <1>       ;MOV   BL,DELAY 1
8100                            <1>                                   ; wait for 10 seconds
8101                            <1>       ;mov   cx, WAIT_HDU_INT_LO ; 1615h
8102                            <1>       ;;mov  bl, WAIT_HDU_INT_HI ;   05h
8103                            <1>       ;mov   bl, WAIT_HDU_INT_HI + 1
8104 00005954 B915160500        <1>       mov    ecx, WAIT_HDU_INT_LH  ; 21/02/2015
8105                            <1>       ;
8106                            <1> ;;      mov     byte [wait_count], 0    ; Reset wait counter
8107                            <1> NB1:
8108 00005959 EC                <1>       IN     AL,DX                ; CHECK STATUS
8109                            <1>       ;TEST  AL,ST_BUSY
8110 0000595A 2480              <1>       and    al, ST_BUSY
8111                            <1>       ;LOOPNZ      NB1
8112 0000595C 7410              <1>       JZ     short NB2            ; NOT BUSY--LETS GO
8113                            <1>       ;DEC   BL
8114                            <1>       ;JNZ   short NB1            ; KEEP TRYING FOR A WHILE
8115                            <1>
8116 0000595E E461              <1> NB1_hi: IN   AL,SYS1                    ; wait for hi to lo
8117 00005960 A810              <1>       TEST   AL,010H                    ; transition on memory
8118 00005962 75FA              <1>       JNZ    SHORT NB1_hi         ; refresh.
8119 00005964 E461              <1> NB1_lo: IN   AL,SYS1
8120 00005966 A810              <1>       TEST   AL,010H
8121 00005968 74FA              <1>       JZ     short NB1_lo
8122 0000596A E2ED              <1>       LOOP   NB1
8123                            <1>       ;dec   bl
8124                            <1>       ;jnz   short NB1
8125                            <1>       ;
8126                            <1> ;;    cmp   byte [wait_count], 182  ; 10 seconds (182 timer ticks)
8127                            <1> ;;    jb     short NB1
8128                            <1>       ;
8129                            <1>       ;MOV   [DISK_STATUS1],TIME_OUT   ; REPORT TIME OUT ERROR
8130                            <1>       ;JMP   SHORT NB3
8131 0000596C B080              <1>       mov    al, TIME_OUT
8132                            <1> NB2:
```

```
                            <1>       ;MOV   byte [DISK_STATUS1],0
                            <1> ;NB3:
                            <1>       ;POP   eBX
0000596E A2[8B890100]       <1>       mov    [DISK_STATUS1], al  ;;; will be set after return
                            <1>       ;CMP   byte [DISK_STATUS1],0 ; SET CONDITION CODE FOR CALLER
00005973 08C0               <1>       or     al, al              ; (zf = 0 --> timeout)
00005975 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
00005976 668B15[C06C0000]   <1>       mov    dx, [HF_PORT]
0000597D 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)
00005980 B9E8030000         <1>       mov    ecx, WAIT_HDU_DRQ_LH ; 21/02/2015
00005985 EC                 <1> WQ_1: IN     AL,DX               ; GET STATUS
00005986 A808               <1>       TEST   AL,ST_DRQ           ; WAIT FOR DRQ
00005988 7516               <1>       JNZ    short WQ_OK
                            <1>       ;LOOP  WQ_1                ; KEEP TRYING FOR A SHORT WHILE
                            <1> WQ_hi:
0000598A E461               <1>       IN     AL,SYS1                   ; wait for hi to lo
0000598C A810               <1>       TEST   AL,010H                   ; transition on memory
0000598E 75FA               <1>       JNZ    SHORT WQ_hi         ; refresh.
00005990 E461               <1> WQ_lo:  IN      AL,SYS1
00005992 A810               <1>       TEST   AL,010H
00005994 74FA               <1>       JZ     SHORT WQ_lo
00005996 E2ED               <1>       LOOP   WQ_1
                            <1>
00005998 C605[8B890100]80   <1>       MOV    byte [DISK_STATUS1],TIME_OUT ; ERROR
0000599F F9                 <1>       STC
                            <1> WQ_OK:
000059A0 C3                 <1>       RETn
                            <1> ;WQ_OK:      ;CLC
                            <1> ;     RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     CHECK FIXED DISK STATUS   :
                            <1> ;----------------------------------------
                            <1> CHECK_STATUS:
000059A1 E813000000         <1>       CALL   CHECK_ST            ; CHECK THE STATUS BYTE
000059A6 7509               <1>       JNZ    short CHECK_S1      ; AN ERROR WAS FOUND
000059A8 A801               <1>       TEST   AL,ST_ERROR         ; WERE THERE ANY OTHER ERRORS
000059AA 7405               <1>       JZ     short CHECK_S1      ; NO ERROR REPORTED
000059AC E849000000         <1>       CALL   CHECK_ER            ; ERROR REPORTED
                            <1> CHECK_S1:
000059B1 803D[8B890100]00   <1>       CMP    byte [DISK_STATUS1],0 ; SET STATUS FOR CALLER
000059B8 C3                 <1>       RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     CHECK FIXED DISK STATUS BYTE     :
                            <1> ;----------------------------------------
                            <1> CHECK_ST:
                            <1>       ;MOV   DX,HF_PORT+7        ; GET THE STATUS
000059B9 668B15[C06C0000]   <1>       mov    dx, [HF_PORT]
000059C0 80C207             <1>       add    dl, 7
                            <1>
                            <1>       ; 17/02/2016
                            <1>       ;(http://wiki.osdev.org/ATA_PIO_Mode)
                            <1>       ;"delay 400ns to allow drive to set new values of BSY and DRQ"
000059C3 EC                 <1>       IN     AL,DX
                            <1>       ;in    al, dx ; 100ns
                            <1>       ;in    al, dx ; 100ns
                            <1>       ;in    al, dx ; 100ns
                            <1>       NEWIODELAY ; 18/02/2016 (AWARD BIOS - 1999, 'CKST' in AHSDK.ASM)
2973 000059C4 E6EB          <2>  out 0ebh,al
                            <1>       ;
000059C6 A2[81890100]       <1>       MOV    [HF_STATUS],AL
000059CB B400               <1>       MOV    AH,0
000059CD A880               <1>       TEST   AL,ST_BUSY          ; IF STILL BUSY
000059CF 751A               <1>       JNZ    short CKST_EXIT     ;  REPORT OK
000059D1 B4CC               <1>       MOV    AH,WRITE_FAULT
000059D3 A820               <1>       TEST   AL,ST_WRT_FLT       ; CHECK FOR WRITE FAULT
000059D5 7514               <1>       JNZ    short CKST_EXIT
000059D7 B4AA               <1>       MOV    AH,NOT_RDY
000059D9 A840               <1>       TEST   AL,ST_READY         ; CHECK FOR NOT READY
000059DB 740E               <1>       JZ     short CKST_EXIT
000059DD B440               <1>       MOV    AH,BAD_SEEK
000059DF A810               <1>       TEST   AL,ST_SEEK_COMPL    ; CHECK FOR SEEK NOT COMPLETE
000059E1 7408               <1>       JZ     short CKST_EXIT
000059E3 B411               <1>       MOV    AH,DATA_CORRECTED
000059E5 A804               <1>       TEST   AL,ST_CORRCTD       ; CHECK FOR CORRECTED ECC
000059E7 7502               <1>       JNZ    short CKST_EXIT
000059E9 B400               <1>       MOV    AH,0
                            <1> CKST_EXIT:
000059EB 8825[8B890100]     <1>       MOV    [DISK_STATUS1],AH   ; SET ERROR FLAG
000059F1 80FC11             <1>       CMP    AH,DATA_CORRECTED   ; KEEP GOING WITH DATA CORRECTED
000059F4 7403               <1>       JZ     short CKST_EX1
000059F6 80FC00             <1>       CMP    AH,0
                            <1> CKST_EX1:
000059F9 C3                 <1>       RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     CHECK FIXED DISK ERROR REGISTER :
                            <1> ;----------------------------------------
                            <1> CHECK_ER:
                            <1>       ;MOV   DX, HF_PORT+1       ; GET THE ERROR REGISTER
000059FA 668B15[C06C0000]   <1>       mov    dx, [HF_PORT]       ;
00005A01 FEC2               <1>       inc    dl
```

```
8237 00005A03 EC                  <1>       IN     AL,DX
8238 00005A04 A2[82890100]        <1>       MOV    [HF_ERROR],AL
8239 00005A09 53                  <1>       PUSH   eBX ; 21/02/2015
8240 00005A0A B908000000          <1>       MOV    eCX,8               ; TEST ALL 8 BITS
8241 00005A0F D0E0                <1> CK1:  SHL    AL,1                ; MOVE NEXT ERROR BIT TO CARRY
8242 00005A11 7202                <1>       JC     short CK2           ; FOUND THE ERROR
8243 00005A13 E2FA                <1>       LOOP   CK1                 ; KEEP TRYING
8244 00005A15 BB[B46C0000]        <1> CK2:  MOV    eBX, ERR_TBL        ; COMPUTE ADDRESS OF
8245 00005A1A 01CB                <1>       ADD    eBX,eCX             ; ERROR CODE
8246                              <1>       ;;MOV  AH,BYTE [CS:BX]     ; GET ERROR CODE
8247                              <1>       ;mov   ah, [bx]
8248 00005A1C 8A23                <1>       mov    ah, [ebx] ; 21/02/2015
8249 00005A1E 8825[8B890100]      <1> CKEX: MOV    [DISK_STATUS1],AH   ; SAVE ERROR CODE
8250 00005A24 5B                  <1>       POP    eBX
8251 00005A25 80FC00              <1>       CMP    AH,0
8252 00005A28 C3                  <1>       RETn
8253                              <1>
8254                              <1> ;--------------------------------------------------------
8255                              <1> ; CHECK_DMA                                        :
8256                              <1> ;  -CHECK ES:BX AND # SECTORS TO MAKE SURE THAT IT WILL :
8257                              <1> ;   FIT WITHOUT SEGMENT OVERFLOW.                    :
8258                              <1> ;  -ES:BX HAS BEEN REVISED TO THE FORMAT SSSS:000X   :
8259                              <1> ;  -OK IF # SECTORS < 80H (7FH IF LONG READ OR WRITE)      :
8260                              <1> ;  -OK IF # SECTORS = 80H (7FH) AND BX <= 00H (04H) :
8261                              <1> ;  -ERROR OTHERWISE                                  :
8262                              <1> ;--------------------------------------------------------
8263                              <1> CHECK_DMA:
8264                              <1>       ; 11/04/2021 (32 bit push/pop)
8265 00005A29 50                  <1>       PUSH   eAX                 ; SAVE REGISTERS
8266 00005A2A 66B80080            <1>       MOV    AX,8000H            ; AH = MAX # SECTORS AL = MAX OFFSET
8267 00005A2E F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE
8268 00005A32 7404                <1>       JZ     short CKD1
8269 00005A34 66B8047F            <1>       MOV    AX,7F04H            ; ECC IS 4 MORE BYTES
8270 00005A38 3A65F9              <1> CKD1: CMP    AH, [CMD_BLOCK+1]   ; NUMBER OF SECTORS
8271 00005A3B 7706                <1>       JA     short CKDOK         ; IT WILL FIT
8272 00005A3D 7207                <1>       JB     short CKDERR        ; TOO MANY
8273 00005A3F 38D8                <1>       CMP    AL,BL               ; CHECK OFFSET ON MAX SECTORS
8274 00005A41 7203                <1>       JB     short CKDERR        ; ERROR
8275 00005A43 F8                  <1> CKDOK: CLC                       ; CLEAR CARRY
8276 00005A44 58                  <1>       POP    eAX
8277 00005A45 C3                  <1>       RETn                        ; NORMAL RETURN
8278 00005A46 F9                  <1> CKDERR: STC                      ; INDICATE ERROR
8279 00005A47 C605[8B890100]09    <1>       MOV    byte [DISK_STATUS1],DMA_BOUNDARY
8280 00005A4E 58                  <1>       POP    eAX
8281 00005A4F C3                  <1>       RETn
8282                              <1>
8283                              <1> ;----------------------------------------
8284                              <1> ;     SET UP ES:BX-> DISK PARMS :
8285                              <1> ;----------------------------------------
8286                              <1>
8287                              <1> ; INPUT -> DL = 0 based drive number
8288                              <1> ; OUTPUT -> ES:BX = disk parameter table address
8289                              <1>
8290                              <1> GET_VEC:
8291                              <1>       ;SUB   AX,AX               ; GET DISK PARAMETER ADDRESS
8292                              <1>       ;MOV   ES,AX
8293                              <1>       ;TEST  DL,1
8294                              <1>       ;JZ    short GV_0
8295                              <1> ;     LES    BX,[HF1_TBL_VEC]    ; ES:BX -> DRIVE PARAMETERS
8296                              <1> ;     JMP    SHORT GV_EXIT
8297                              <1> ;GV_0:
8298                              <1> ;     LES    BX,[HF_TBL_VEC]           ; ES:BX -> DRIVE PARAMETERS
8299                              <1> ;
8300                              <1>       ;xor   bh, bh
8301 00005A50 31DB                <1>       xor    ebx, ebx
8302 00005A52 88D3                <1>       mov    bl, dl
8303                              <1>       ;;02/01/2015
8304                              <1>       ;;shl  bl, 1               ; port address offset
8305                              <1>       ;;mov  ax, [bx+hd_ports]   ; Base port address (1F0h, 170h)
8306                              <1>       ;;shl  bl, 1               ; dpt pointer offset
8307 00005A54 C0E302              <1>       shl    bl, 2  ;;
8308                              <1>       ;add   bx, HF_TBL_VEC            ; Disk parameter table pointer
8309 00005A57 81C3[90890100]      <1>       add    ebx, HF_TBL_VEC ; 21/02/2015
8310                              <1>       ;push  word [bx+2]         ; dpt segment
8311                              <1>       ;pop   es
8312                              <1>       ;mov   bx, [bx]            ; dpt offset
8313 00005A5D 8B1B                <1>       mov    ebx, [ebx]
8314                              <1> ;GV_EXIT:
8315 00005A5F C3                  <1>       RETn
8316                              <1>
8317                              <1> hdc1_int: ; 21/02/2015
8318                              <1> ;--- HARDWARE INT 76H -- ( IRQ LEVEL  14 ) ----------------------
8319                              <1> ;                                                  :
8320                              <1> ;     FIXED DISK INTERRUPT ROUTINE                       :
8321                              <1> ;                                                  :
8322                              <1> ;--------------------------------------------------------------
8323                              <1>
8324                              <1> ; 22/12/2014
8325                              <1> ; IBM PC-XT Model 286 System BIOS Source Code - DISK.ASM (HD_INT)
8326                              <1> ;      '11/15/85'
8327                              <1> ; AWARD BIOS 1999 (D1A0622)
8328                              <1> ;     Source Code - ATORGS.ASM (INT_HDISK, INT_HDISK1)
8329                              <1>
8330                              <1> ;int_76h:
8331                              <1> HD_INT:
8332                              <1>       ; 11/04/2021 (32 bit push/pop)
8333 00005A60 50                  <1>       PUSH   eAX
8334 00005A61 1E                  <1>       PUSH   DS
8335                              <1>       ;CALL  DDS
8336                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
8337 00005A62 66B81000            <1>       mov    ax, KDATA
8338 00005A66 8ED8                <1>       mov    ds, ax
8339                              <1>       ;
8340                              <1>       ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
8341                              <1>       ;mov   byte [CS:HF_INT_FLAG], 0FFh
```

```
8342 00005A68 C605[83890100]FF    <1>       mov    byte [HF_INT_FLAG], 0FFh
8343                              <1>       ;
8344 00005A6F 52                  <1>       push   Edx
8345 00005A70 66BAF701            <1>       mov    dx, HDC1_BASEPORT+7 ; Status Register (1F7h)
8346                              <1>                                  ; Clear Controller
8347                              <1> Clear_IRQ1415:                        ; (Award BIOS - 1999)
8348 00005A74 EC                  <1>       in     al, dx              ;
8349 00005A75 5A                  <1>       pop    Edx
8350                              <1>       NEWIODELAY
2973 00005A76 E6EB                <2>  out 0ebh,al
8351                              <1>       ;
8352 00005A78 B020                <1>       MOV    AL,EOI              ; NON-SPECIFIC END OF INTERRUPT
8353 00005A7A E6A0                <1>       OUT    INTB00,AL           ; FOR CONTROLLER #2
8354                              <1>       ;JMP   $+2                 ; WAIT
8355                              <1>       NEWIODELAY
2973 00005A7C E6EB                <2>  out 0ebh,al
8356 00005A7E E620                <1>       OUT    INTA00,AL           ; FOR CONTROLLER #1
8357 00005A80 1F                  <1>       POP    DS
8358                              <1>       ;STI                       ; RE-ENABLE INTERRUPTS
8359                              <1>       ;MOV   AX,9100H            ; DEVICE POST
8360                              <1>       ;INT   15H                 ;  INTERRUPT
8361                              <1> irq15_iret: ; 25/02/2015
8362 00005A81 58                  <1>       POP    eAX
8363 00005A82 CF                  <1>       IRETd                      ; RETURN FROM INTERRUPT
8364                              <1>
8365                              <1> hdc2_int: ; 21/02/2015
8366                              <1> ;++++ HARDWARE INT 77H ++ ( IRQ LEVEL  15 ) ++++++++++++++++++++++
8367                              <1> ;                                                      :
8368                              <1> ;     FIXED DISK INTERRUPT ROUTINE                           :
8369                              <1> ;                                                      :
8370                              <1> ;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
8371                              <1>
8372                              <1> ;int 77h:
8373                              <1> HD1_INT:
8374                              <1>       ; 11/04/2021 (32 bit push/pop)
8375 00005A83 50                  <1>       PUSH   eAX
8376                              <1>       ; Check if that is a spurious IRQ (from slave PIC)
8377                              <1>       ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
8378 00005A84 B00B                <1>       mov    al, 0Bh  ; In-Service Register
8379 00005A86 E6A0                <1>       out    0A0h, al
8380 00005A88 EB00                <1>         jmp short $+2
8381 00005A8A EB00                <1>       jmp short $+2
8382 00005A8C E4A0                <1>       in     al, 0A0h
8383 00005A8E 2480                <1>       and    al, 80h ; bit 7 (is it real IRQ 15 or fake?)
8384 00005A90 74EF                <1>       jz     short irq15_iret ; Fake (spurious) IRQ, do not send EOI)
8385                              <1>       ;
8386 00005A92 1E                  <1>       PUSH   DS
8387                              <1>       ;CALL  DDS
8388                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
8389 00005A93 66B81000            <1>       mov    ax, KDATA
8390 00005A97 8ED8                <1>       mov    ds, ax
8391                              <1>       ;
8392                              <1>       ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
8393                              <1>         ;or  byte [CS:HF_INT_FLAG],0C0h
8394 00005A99 800D[83890100]C0    <1>       or     byte [HF_INT_FLAG], 0C0h
8395                              <1>       ;
8396 00005AA0 52                  <1>       push   Edx
8397 00005AA1 66BA7701            <1>       mov    dx, HDC2_BASEPORT+7 ; Status Register (177h)
8398                              <1>                                  ; Clear Controller (Award BIOS 1999)
8399 00005AA5 EBCD                <1>       jmp    short Clear_IRQ1415
8400                              <1>
8401                              <1> ;%include 'diskdata.inc' ; 11/03/2015
8402                              <1> ;%include 'diskbss.inc' ; 11/03/2015
8403                              <1>
8404                              <1> ;////////////////////////////////////////////////////////////////
8405                              <1> ;; END OF DISK I/O SYTEM ///
2892                                  %include 'memory.s'  ; 09/03/2015
2893                              <1> ; ============================================================================
2894                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - memory.s
2895                              <1> ; ----------------------------------------------------------------------------
2896                              <1> ; Last Update: 15/12/2020
2897                              <1> ; ----------------------------------------------------------------------------
2898                              <1> ; Beginning: 24/01/2016
2899                              <1> ; ----------------------------------------------------------------------------
2900                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
2901                              <1> ; ----------------------------------------------------------------------------
2902                              <1> ; Turkish Rational DOS
2903                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
2904                              <1> ;
2905                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
2906                              <1> ; memory.inc (18/10/2015)
2907                              <1> ; ============================================================================
2908                              <1>
2909                              <1> ; MEMORY.ASM - Retro UNIX 386 v1 MEMORY MANAGEMENT FUNCTIONS (PROCEDURES)
2910                              <1> ; Retro UNIX 386 v1 Kernel (unix386.s, v0.2.0.14) - MEMORY.INC
2911                              <1> ; Last Modification: 18/10/2015
2912                              <1>
2913                              <1> ; ///////// MEMORY MANAGEMENT FUNCTIONS (PROCEDURES) ///////////////
2914                              <1>
2915                              <1> ;;04/11/2014 (unix386.s)
2916                              <1> ;PDE_A_PRESENT equ 1            ; Present flag for PDE
2917                              <1> ;PDE_A_WRITE equ   2            ; Writable (write permission) flag
2918                              <1> ;PDE_A_USER  equ   4            ; User (non-system/kernel) page flag
2919                              <1> ;;
2920                              <1> ;PTE_A_PRESENT equ 1            ; Present flag for PTE (bit 0)
2921                              <1> ;PTE_A_WRITE equ   2            ; Writable (write permission) flag (bit 1)
2922                              <1> ;PTE_A_USER  equ   4            ; User (non-system/kernel) page flag (bit 2)
2923                              <1> ;PTE_A_ACCESS equ  32           ; Accessed flag (bit 5) ; 09/03/2015
2924                              <1>
2925                              <1> ; 27/04/2015
2926                              <1> ; 09/03/2015
2927                              <1> PAGE_SIZE   equ 4096            ; page size in bytes
2928                              <1> PAGE_SHIFT  equ 12              ; page table shift count
2929                              <1> PAGE_D_SHIFT equ 22 ; 12 + 10   ; page directory shift count
2930                              <1> PAGE_OFF    equ 0FFFh           ; 12 bit byte offset in page frame
```

```
2931                    <1> PTE_MASK     equ 03FFh          ; page table entry mask
2932                    <1> PTE_DUPLICATED  equ 200h        ; duplicated page sign (AVL bit 0)
2933                    <1> PDE_A_CLEAR  equ 0F000h         ; to clear PDE attribute bits
2934                    <1> PTE_A_CLEAR  equ 0F000h         ; to clear PTE attribute bits
2935                    <1> LOGIC_SECT_SIZE equ 512         ; logical sector size
2936                    <1> ERR_MAJOR_PF equ 0E0h           ; major error: page fault
2937                    <1> ERR_MINOR_IM equ 4 ;15/10/2016 (1->4); insufficient (out of) memory
2938                    <1> ERR_MINOR_PV equ 6 ;15/10/2016 (1->4); protection violation
2939                    <1> SWP_DISK_READ_ERR     equ 40
2940                    <1> SWP_DISK_NOT_PRESENT_ERR   equ 41
2941                    <1> SWP_SECTOR_NOT_PRESENT_ERR equ 42
2942                    <1> SWP_NO_FREE_SPACE_ERR      equ 43
2943                    <1> SWP_DISK_WRITE_ERR         equ 44
2944                    <1> SWP_NO_PAGE_TO_SWAP_ERR    equ 45
2945                    <1> PTE_A_ACCESS_BIT equ 5  ; Bit 5 (accessed flag)
2946                    <1> SECTOR_SHIFT     equ 3  ; sector shift (to convert page block number)
2947                    <1> ; 12/07/2016
2948                    <1> PTE_SHARED   equ 400h           ; AVL bit 1, direct memory access bit
2949                    <1>                                 ; (Indicates that the page is not allocated
2950                    <1>                                 ; for the process, it is a shared or system
2951                    <1>                                 ; page, it must not be deallocated!)
2952                    <1> ; 14/12/2020
2953                    <1> ; (Linear Frame Buffer - video memory mark : AVL bit 1, outside M.A.T.)
2954                    <1> PDE_EXTERNAL equ 400h     ; Page directory entry for external memory blocks
2955                    <1> PTE_EXTERNAL equ 400h     ; Allocated kernel pages for Linear Frame Buffer
2956                    <1>                           ; (Out of memory allocation table)
2957                    <1>
2958                    <1> ;
2959                    <1> ;; Retro Unix 386 v1 - paging method/principles
2960                    <1> ;;
2961                    <1> ;; 10/10/2014
2962                    <1> ;; RETRO UNIX 386 v1 - PAGING METHOD/PRINCIPLES
2963                    <1> ;;
2964                    <1> ;; KERNEL PAGE MAP: 1 to 1 physical memory page map
2965                    <1> ;;    (virtual address = physical address)
2966                    <1> ;; KERNEL PAGE TABLES:
2967                    <1> ;;    Kernel page directory and all page tables are
2968                    <1> ;;    on memory as initialized, as equal to physical memory
2969                    <1> ;;    layout. Kernel pages can/must not be swapped out/in.
2970                    <1> ;;
2971                    <1> ;;    what for: User pages may be swapped out, when accessing
2972                    <1> ;;    a page in kernel/system mode, if it would be swapped out,
2973                    <1> ;;    kernel would have to swap it in! But it is also may be
2974                    <1> ;;    in use by a user process. (In system/kernel mode
2975                    <1> ;;    kernel can access all memory pages even if they are
2976                    <1> ;;    reserved/allocated for user processes. Swap out/in would
2977                    <1> ;;    cause conflicts.)
2978                    <1> ;;
2979                    <1> ;;    As result of these conditions,
2980                    <1> ;;    all kernel pages must be initialized as equal to
2981                    <1> ;;    physical layout for preventing page faults.
2982                    <1> ;;    Also, calling "allocate page" procedure after
2983                    <1> ;;    a page fault can cause another page fault (double fault)
2984                    <1> ;;    if all kernel page tables would not be initialized.
2985                    <1> ;;
2986                    <1> ;;    [first_page] = Beginning of users space, as offset to
2987                    <1> ;;    memory allocation table. (double word aligned)
2988                    <1> ;;
2989                    <1> ;;    [next_page] = first/next free space to be searched
2990                    <1> ;;    as offset to memory allocation table. (dw aligned)
2991                    <1> ;;
2992                    <1> ;;    [last_page] = End of memory (users space), as offset
2993                    <1> ;;    to memory allocation table. (double word aligned)
2994                    <1> ;;
2995                    <1> ;; USER PAGE TABLES:
2996                    <1> ;;    Demand paging (& 'copy on write' allocation method) ...
2997                    <1> ;;           'ready only' marked copies of the
2998                    <1> ;;           parent process's page table entries (for
2999                    <1> ;;           same physical memory).
3000                    <1> ;;           (A page will be copied to a new page after
3001                    <1> ;;            if it causes R/W page fault.)
3002                    <1> ;;
3003                    <1> ;;    Every user process has own (different)
3004                    <1> ;;    page directory and page tables.
3005                    <1> ;;
3006                    <1> ;;    Code starts at virtual address 0, always.
3007                    <1> ;;    (Initial value of EIP is 0 in user mode.)
3008                    <1> ;;    (Programs can be written/developed as simple
3009                    <1> ;;     flat memory programs.)
3010                    <1> ;;
3011                    <1> ;; MEMORY ALLOCATION STRATEGY:
3012                    <1> ;;    Memory page will be allocated by kernel only
3013                    <1> ;;           (in kernel/system mode only).
3014                    <1> ;;    * After a
3015                    <1> ;;      - 'not present' page fault
3016                    <1> ;;      - 'writing attempt on read only page' page fault
3017                    <1> ;;    * For loading (opening, reading) a file or disk/drive
3018                    <1> ;;    * As responce to 'allocate additional memory blocks'
3019                    <1> ;;      request by running process.
3020                    <1> ;;    * While creating a process, allocating a new buffer,
3021                    <1> ;;      new page tables etc.
3022                    <1> ;;
3023                    <1> ;;    At first,
3024                    <1> ;;    - 'allocate page' procedure will be called;
3025                    <1> ;,       if it will return with a valid (>0) physical address
3026                    <1> ;;       (that means the relevant M.A.T. bit has been RESET)
3027                    <1> ;;       relevant memory page/block will be cleared (zeroed).
3028                    <1> ;;    - 'allocate page' will be called for allocating page
3029                    <1> ;;       directory, page table and running space (data/code).
3030                    <1> ;;    - every successful 'allocate page' call will decrease
3031                    <1> ;;      'free_pages' count (pointer).
3032                    <1> ;;    - 'out of (insufficient) memory error' will be returned
3033                    <1> ;;      if 'free_pages' points to a ZERO.
3034                    <1> ;;    - swapping out and swapping in (if it is not a new page)
3035                    <1> ;;      procedures will be called as responce to 'out of memory'
```

```
3036                    <1> ;;      error except errors caused by attribute conflicts.
3037                    <1> ;;     (swapper functions)
3038                    <1> ;;
3039                    <1> ;;    At second,
3040                    <1> ;;    - page directory entry will be updated then page table
3041                    <1> ;;      entry will be updated.
3042                    <1> ;;
3043                    <1> ;; MEMORY ALLOCATION TABLE FORMAT:
3044                    <1> ;;    - M.A.T. has a size according to available memory as
3045                    <1> ;;      follows:
3046                    <1> ;;            - 1 (allocation) bit per 1 page (4096 bytes)
3047                    <1> ;;            - a bit with value of 0 means allocated page
3048                    <1> ;;            - a bit with value of 1 means a free page
3049                    <1> ;,    - 'free_pages' pointer holds count of free pages
3050                    <1> ;;      depending on M.A.T.
3051                    <1> ;;          (NOTE: Free page count will not be checked
3052                    <1> ;;          again -on M.A.T.- after initialization.
3053                    <1> ;;          Kernel will trust on initial count.)
3054                    <1> ;,    - 'free_pages' count will be decreased by allocation
3055                    <1> ;;      and it will be increased by deallocation procedures.
3056                    <1> ;;
3057                    <1> ;;    - Available memory will be calculated during
3058                    <1> ;;      the kernel's initialization stage (in real mode).
3059                    <1> ;;      Memory allocation table and kernel page tables
3060                    <1> ;;      will be formatted/sized as result of available
3061                    <1> ;;      memory calculation before paging is enabled.
3062                    <1> ;;
3063                    <1> ;; For 4GB Available/Present Memory: (max. possible memory size)
3064                    <1> ;;    - Memory Allocation Table size will be 128 KB.
3065                    <1> ;;    - Memory allocation for kernel page directory size
3066                    <1> ;;      is always 4 KB. (in addition to total allocation size
3067                    <1> ;;      for page tables)
3068                    <1> ;;    - Memory allocation for kernel page tables (1024 tables)
3069                    <1> ;;      is 4 MB (1024*4*1024 bytes).
3070                    <1> ;;    - User (available) space will be started
3071                    <1> ;;      at 6th MB of the memory (after 1MB+4MB).
3072                    <1> ;;    - The first 640 KB is for kernel's itself plus
3073                    <1> ;;      memory allocation table and kernel's page directory
3074                    <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
3075                    <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
3076                    <1> ;;      for buffers.
3077                    <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
3078                    <1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
3079                    <1> ;;    - Kernel page tables start at 100000h (2nd MB)
3080                    <1> ;;
3081                    <1> ;; For 1GB Available Memory:
3082                    <1> ;;    - Memory Allocation Table size will be 32 KB.
3083                    <1> ;;    - Memory allocation for kernel page directory size
3084                    <1> ;;      is always 4 KB. (in addition to total allocation size
3085                    <1> ;;      for page tables)
3086                    <1> ;;    - Memory allocation for kernel page tables (256 tables)
3087                    <1> ;;      is 1 MB (256*4*1024 bytes).
3088                    <1> ;;    - User (available) space will be started
3089                    <1> ;;      at 3th MB of the memory (after 1MB+1MB).
3090                    <1> ;;    - The first 640 KB is for kernel's itself plus
3091                    <1> ;;      memory allocation table and kernel's page directory
3092                    <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
3093                    <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
3094                    <1> ;;      for buffers.
3095                    <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
3096                    <1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
3097                    <1> ;;    - Kernel page tables start at 100000h (2nd MB).
3098                    <1> ;;
3099                    <1> ;;
3100                    <1>
3101                    <1>
3102                    <1> ;;******************************************************************************
3103                    <1> ;;
3104                    <1> ;; RETRO UNIX 386 v1 - Paging (Method for Copy On Write paging principle)
3105                    <1> ;; DEMAND PAGING - PARENT&CHILD PAGE TABLE DUPLICATION PRINCIPLES (23/04/2015)
3106                    <1>
3107                    <1> ;; Main factor: "sys fork" system call
3108                    <1> ;;
3109                    <1> ;;          FORK
3110                    <1> ;;                      |----> parent - duplicated PTEs, read only pages
3111                    <1> ;;  writable pages ---->|
3112                    <1> ;;                      |----> child - duplicated PTEs, read only pages
3113                    <1> ;;
3114                    <1> ;; AVL bit (0) of Page Table Entry is used as duplication sign
3115                    <1> ;;
3116                    <1> ;; AVL Bit 0 [PTE Bit 9] = 'Duplicated PTE belongs to child' sign/flag (if it is set)
3117                    <1> ;; Note: Dirty bit (PTE bit 6) may be used instead of AVL bit 0 (PTE bit 9)
3118                    <1> ;;       -while R/W bit is 0-.
3119                    <1> ;;
3120                    <1> ;; Duplicate page tables with writable pages (the 1st sys fork in the process):
3121                    <1> ;; # Parent's Page Table Entries are updated to point same pages as read only,
3122                    <1> ;;   as duplicated PTE bit  -AVL bit 0, PTE bit 9- are reset/clear.
3123                    <1> ;; # Then Parent's Page Table is copied to Child's Page Table.
3124                    <1> ;; # Child's Page Table Entries are updated as duplicated child bit
3125                    <1> ;;   -AVL bit 0, PTE bit 9- is set.
3126                    <1> ;;
3127                    <1> ;; Duplicate page tables with read only pages (several sys fork system calls):
3128                    <1> ;; # Parent's read only pages are copied to new child pages.
3129                    <1> ;;   Parent's PTE attributes are not changed.
3130                    <1> ;;   (Because, there is another parent-child fork before this fork! We must not
3131                    <1> ;;    destroy/mix previous fork result).
3132                    <1> ;; # Child's Page Table Entries (which are corresponding to Parent's
3133                    <1> ;;   read only pages) are set as writable (while duplicated PTE bit is clear).
3134                    <1> ;; # Parent's PTEs with writable page attribute are updated to point same pages
3135                    <1> ;;   as read only, (while) duplicated PTE bit is reset (clear).
3136                    <1> ;; # Parent's Page Table Entries (with writable page attribute) are duplicated
3137                    <1> ;;   as Child's Page Table Entries without copying actual page.
3138                    <1> ;; # Child 's Page Table Entries (which are corresponding to Parent's writable
3139                    <1> ;;   pages) are updated as duplicated PTE bit (AVL bit 0, PTE bit 9- is set.
3140                    <1> ;;
```

```
                                    <1> ;; !? WHAT FOR (duplication after duplication):
                                    <1> ;; In UNIX method for sys fork (a typical 'fork' application in /etc/init)
                                    <1> ;; program/executable code continues from specified location as child process,
                                    <1> ;; returns back previous code location as parent process, every child after
                                    <1> ;; every sys fork uses last image of code and data just prior the fork.
                                    <1> ;; Even if the parent code changes data, the child will not see the changed data
                                    <1> ;; after the fork. In Retro UNIX 8086 v1, parent's process segment (32KB)
                                    <1> ;; was copied to child's process segment (all of code and data) according to
                                    <1> ;; original UNIX v1 which copies all of parent process code and data -core-
                                    <1> ;; to child space -core- but swaps that core image -of child- on to disk.
                                    <1> ;; If I (Erdogan Tan) would use a method of to copy parent's core
                                    <1> ;; (complete running image of parent process) to the child process;
                                    <1> ;; for big sizes, i would force Retro UNIX 386 v1 to spend many memory pages
                                    <1> ;; and times only for a sys fork. (It would excessive reservation for sys fork,
                                    <1> ;; because sys fork usually is prior to sys exec; sys exec always establishes
                                    <1> ;; a new/fresh core -running space-, by clearing all code/data content).
                                    <1> ;; 'Read Only' page flag ensures page fault handler is needed only for a few write
                                    <1> ;; attempts between sys fork and sys exec, not more... (I say so by thinking
                                    <1> ;; of "/etc/init" content, specially.) sys exec will clear page tables and
                                    <1> ;; new/fresh pages will be used to load and run new executable/program.
                                    <1> ;; That is what for i have preferred "copy on write", "duplication" method
                                    <1> ;; for sharing same read only pages between parent and child processes.
                                    <1> ;; That is a pitty i have to use new private flag (AVL bit 0, "duplicated PTE
                                    <1> ;; belongs to child" sign) for cooperation on duplicated pages between a parent
                                    <1> ;; and it's child processes; otherwise parent process would destroy data belongs
                                    <1> ;; to its child or vice versa; or some pages would remain unclaimed
                                    <1> ;; -deallocation problem-.
                                    <1> ;; Note: to prevent conflicts, read only pages must not be swapped out...
                                    <1> ;;
                                    <1> ;; WHEN PARENT TRIES TO WRITE IT'S READ ONLY (DUPLICATED) PAGE:
                                    <1> ;; # Page fault handler will do those:
                                    <1> ;;   - 'Duplicated PTE' flag (PTE bit 9) is checked (on the failed PTE).
                                    <1> ;;   - If it is reset/clear, there is a child uses same page.
                                    <1> ;;   - Parent's read only page -previous page- is copied to a new writable page.
                                    <1> ;;   - Parent's PTE is updated as writable page, as unique page (AVL=0)
                                    <1> ;;   - (Page fault handler whill check this PTE later, if child process causes to
                                    <1> ;;     page fault due to write attempt on read only page. Of course, the previous
                                    <1> ;;     read only page will be converted to writable and unique page which belongs
                                    <1> ;;     to child process.)
                                    <1> ;; WHEN CHILD TRIES TO WRITE IT'S READ ONLY (DUPLICATED) PAGE:
                                    <1> ;; # Page fault handler will do those:
                                    <1> ;;   - 'Duplicated PTE' flag (PTE bit 9) is checked (on the failed PTE).
                                    <1> ;;   - If it is set, there is a parent uses -or was using- same page.
                                    <1> ;;   - Same PTE address within parent's page table is checked if it has same page
                                    <1> ;;     address or not.
                                    <1> ;;   - If parent's PTE has same address, child will continue with a new writable page.
                                    <1> ;;     Parent's PTE will point to same (previous) page as writable, unique (AVL=0).
                                    <1> ;;   - If parent's PTE has different address, child will continue with it's
                                    <1> ;;     own/same page but read only flag (0) will be changed to writable flag (1) and
                                    <1> ;;     'duplicated PTE (belongs to child)' flag/sign will be cleared/reset.
                                    <1> ;;
                                    <1> ;; NOTE: When a child process is terminated, read only flags of parent's page tables
                                    <1> ;;       will be set as writable (and unique) in case of child process was using
                                    <1> ;;       same pages with duplicated child PTE sign... Depending on sys fork and
                                    <1> ;;       duplication method details, it is not possible multiple child processes
                                    <1> ;;       were using same page with duplicated PTEs.
                                    <1> ;;
                                    <1> ;;////////////////////////////////////////////////////////////////////////////////
                                    <1>
                                    <1> ;; 08/10/2014
                                    <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>       ; 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>       ;
00005AA7 A1[F8880100]               <1>       mov   eax, [free_pages]
00005AAC 21C0                       <1>       and   eax, eax
00005AAE 7438                       <1>       jz    short out_of_memory
                                    <1>       ;
00005AB0 53                         <1>       push  ebx
00005AB1 51                         <1>       push  ecx
                                    <1>       ;
00005AB2 BB00001000                 <1>       mov   ebx, MEM_ALLOC_TBL   ; Memory Allocation Table offset
00005AB7 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)
00005AB9 031D[FC880100]             <1>       add   ebx, [next_page] ; Free page searching starts from here
                                    <1>                              ; next_free_page >> 5
00005ABF 030D[00890100]             <1>       add   ecx, [last_page] ; Free page searching ends here
```

```
                        <1>                             ; (total_pages - 1) >> 5
                        <1> al_p_scan:
00005AC5 39CB           <1>       cmp   ebx, ecx
00005AC7 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>       ;
00005AC9 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.
00005ACC 7525           <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!)
00005ACE 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.
00005AD1 EBF2           <1>         jmp     short al_p_scan
                        <1>       ;
                        <1> al_p_notfound:
00005AD3 81E900001000   <1>       sub   ecx, MEM_ALLOC_TBL
00005AD9 890D[FC880100] <1>       mov   [next_page], ecx ; next/first free page = last page
                        <1>                              ; (deallocate_page procedure will change it)
00005ADF 31C0           <1>       xor   eax, eax
00005AE1 A3[F8880100]   <1>       mov   [free_pages], eax ; 0
00005AE6 59             <1>       pop   ecx
00005AE7 5B             <1>       pop   ebx
                        <1>       ;
                        <1> out_of_memory:
00005AE8 E85B040000     <1>       call  swap_out
00005AED 7325           <1>       jnc   short al_p_ok  ; [free_pages] = 0, re-allocation by swap_out
                        <1>       ;
00005AEF 29C0           <1>       sub   eax, eax ; 0
00005AF1 F9             <1>       stc
00005AF2 C3             <1>       retn
                        <1>
                        <1> al_p_found:
00005AF3 89D9           <1>       mov   ecx, ebx
00005AF5 81E900001000   <1>       sub   ecx, MEM_ALLOC_TBL
00005AFB 890D[FC880100] <1>       mov   [next_page], ecx ; Set first free page searching start
                        <1>                              ; address/offset (to the next)
00005B01 FF0D[F8880100] <1>         dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
                        <1>       ;
00005B07 0FB303         <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 page.
                        <1>       ; 01/07/2015 (4*8 = 32, 1 allocation byte = 8 pages)
00005B0A C1E103         <1>       shl   ecx, 3        ; (page block offset * 32) + page index
00005B0D 01C8           <1>       add   eax, ecx      ; = page number
00005B0F C1E00C         <1>       shl   eax, 12             ; physical address of the page (flat/real value)
                        <1>       ; EAX = physical address of memory page
                        <1>       ;
                        <1>       ; NOTE: The relevant page directory and page table entry will be updated
                        <1>       ;       according to this EAX value...
00005B12 59             <1>       pop   ecx
00005B13 5B             <1>       pop   ebx
                        <1> al_p_ok:
00005B14 C3             <1>       retn
                        <1>
                        <1>
                        <1> make_page_dir:
                        <1>       ; 18/04/2015
                        <1>       ; 12/04/2015
                        <1>       ; 23/10/2014
                        <1>       ; 16/10/2014
                        <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;     none
                        <1>       ; OUTPUT ->
                        <1>       ;     (EAX = 0)
                        <1>       ;     cf = 1 -> insufficient (out of) memory error
                        <1>       ;     cf = 0 ->
                        <1>       ;     u.pgdir = page directory (physical) address of the current
                        <1>       ;             process/user.
                        <1>       ;
                        <1>       ; Modified Registers -> EAX
                        <1>       ;
00005B15 E88DFFFFFF     <1>       call  allocate_page
00005B1A 7216           <1>       jc    short mkpd_error
                        <1>       ;
00005B1C A3[B8030300]   <1>       mov   [u.pgdir], eax    ; Page dir address for current user/process
                        <1>                               ; (Physical address)
```

```
3351                            <1> clear_page:
3352                            <1>       ; 18/04/2015
3353                            <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
3354                            <1>       ;
3355                            <1>       ; INPUT ->
3356                            <1>       ;     EAX = physical address of the page
3357                            <1>       ; OUTPUT ->
3358                            <1>       ;     all bytes of the page will be cleared
3359                            <1>       ;
3360                            <1>       ; Modified Registers -> none
3361                            <1>       ;
3362 00005B21 57               <1>       push   edi
3363 00005B22 51               <1>       push   ecx
3364 00005B23 50               <1>       push   eax
3365 00005B24 B900040000       <1>       mov    ecx, PAGE_SIZE / 4
3366 00005B29 89C7             <1>       mov    edi, eax
3367 00005B2B 31C0             <1>       xor    eax, eax
3368 00005B2D F3AB             <1>       rep    stosd
3369 00005B2F 58               <1>       pop    eax
3370 00005B30 59               <1>       pop    ecx
3371 00005B31 5F               <1>       pop    edi
3372                            <1> mkpd_error:
3373                            <1> mkpt_error:
3374 00005B32 C3               <1>       retn
3375                            <1>
3376                            <1> make_page_table:
3377                            <1>       ; 23/06/2015
3378                            <1>       ; 18/04/2015
3379                            <1>       ; 12/04/2015
3380                            <1>       ; 16/10/2014
3381                            <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
3382                            <1>       ;
3383                            <1>       ; INPUT ->
3384                            <1>       ;     EBX = virtual (linear) address
3385                            <1>       ;     ECX = page table attributes (lower 12 bits)
3386                            <1>       ;           (higher 20 bits must be ZERO)
3387                            <1>       ;           (bit 0 must be 1)
3388                            <1>       ;     u.pgdir = page directory (physical) address
3389                            <1>       ; OUTPUT ->
3390                            <1>       ;     EDX = Page directory entry address
3391                            <1>       ;     EAX = Page table address
3392                            <1>       ;     cf = 1 -> insufficient (out of) memory error
3393                            <1>       ;     cf = 0 -> page table address in the PDE (EDX)
3394                            <1>       ;
3395                            <1>       ; Modified Registers -> EAX, EDX
3396                            <1>       ;
3397 00005B33 E86FFFFFFF       <1>       call   allocate_page
3398 00005B38 72F8             <1>       jc     short mkpt_error
3399 00005B3A E811000000       <1>       call   set_pde
3400 00005B3F EBE0             <1>       jmp    short clear_page
3401                            <1>
3402                            <1> make_page:
3403                            <1>       ; 24/07/2015
3404                            <1>       ; 23/06/2015 ; (Retro UNIX 386 v1 - beginning)
3405                            <1>       ;
3406                            <1>       ; INPUT ->
3407                            <1>       ;     EBX = virtual (linear) address
3408                            <1>       ;     ECX = page attributes (lower 12 bits)
3409                            <1>       ;           (higher 20 bits must be ZERO)
3410                            <1>       ;           (bit 0 must be 1)
3411                            <1>       ;     u.pgdir = page directory (physical) address
3412                            <1>       ; OUTPUT ->
3413                            <1>       ;     EBX = Virtual address
3414                            <1>       ;     (EDX = PTE value)
3415                            <1>       ;     EAX = Physical address
3416                            <1>       ;     cf = 1 -> insufficient (out of) memory error
3417                            <1>       ;
3418                            <1>       ; Modified Registers -> EAX, EDX
3419                            <1>       ;
3420 00005B41 E861FFFFFF       <1>       call   allocate_page
3421 00005B46 7207             <1>       jc     short mkp_err
3422 00005B48 E821000000       <1>       call   set_pte
3423 00005B4D 73D2             <1>       jnc    short clear_page ; 18/04/2015
3424                            <1> mkp_err:
3425 00005B4F C3               <1>       retn
3426                            <1>
3427                            <1>
3428                            <1> set_pde:    ; Set page directory entry (PDE)
3429                            <1>       ; 20/07/2015
3430                            <1>       ; 18/04/2015
3431                            <1>       ; 12/04/2015
3432                            <1>       ; 23/10/2014
3433                            <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
3434                            <1>       ;
3435                            <1>       ; INPUT ->
3436                            <1>       ;     EAX = physical address
3437                            <1>       ;           (use present value if EAX = 0)
3438                            <1>       ;     EBX = virtual (linear) address
3439                            <1>       ;     ECX = page table attributes (lower 12 bits)
3440                            <1>       ;           (higher 20 bits must be ZERO)
3441                            <1>       ;           (bit 0 must be 1)
3442                            <1>       ;     u.pgdir = page directory (physical) address
3443                            <1>       ; OUTPUT ->
3444                            <1>       ;     EDX = PDE address
3445                            <1>       ;     EAX = page table address (physical)
3446                            <1>       ;     ;(CF=1 -> Invalid page address)
3447                            <1>       ;
3448                            <1>       ; Modified Registers -> EDX
3449                            <1>       ;
3450 00005B50 89DA             <1>       mov    edx, ebx
3451 00005B52 C1EA16           <1>       shr    edx, PAGE_D_SHIFT ; 22
3452 00005B55 C1E202           <1>       shl    edx, 2 ; offset to page directory (1024*4)
3453 00005B58 0315[B8030300]   <1>       add    edx, [u.pgdir]
3454                            <1>       ;
3455 00005B5E 21C0             <1>       and    eax, eax
```

```
3456 00005B60 7506                <1>       jnz    short spde_1
3457                              <1>       ;
3458 00005B62 8B02                <1>       mov    eax, [edx]  ; old PDE value
3459                              <1>       ;test  al, 1
3460                              <1>       ;jz    short spde_2
3461 00005B64 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h  ; clear lower 12 bits
3462                              <1> spde_1:
3463                              <1>       ;and   cx, 0FFFh
3464 00005B68 8902                <1>       mov    [edx], eax
3465 00005B6A 66090A              <1>       or     [edx], cx
3466 00005B6D C3                  <1>       retn
3467                              <1> ;spde_2: ; error
3468                              <1> ;     stc
3469                              <1> ;     retn
3470                              <1>
3471                              <1> set_pte:    ; Set page table entry (PTE)
3472                              <1>       ; 24/07/2015
3473                              <1>       ; 20/07/2015
3474                              <1>       ; 23/06/2015
3475                              <1>       ; 18/04/2015
3476                              <1>       ; 12/04/2015
3477                              <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
3478                              <1>       ;
3479                              <1>       ; INPUT ->
3480                              <1>       ;     EAX = physical page address
3481                              <1>       ;           (use present value if EAX = 0)
3482                              <1>       ;     EBX = virtual (linear) address
3483                              <1>       ;     ECX = page attributes (lower 12 bits)
3484                              <1>       ;           (higher 20 bits must be ZERO)
3485                              <1>       ;           (bit 0 must be 1)
3486                              <1>       ;     u.pgdir = page directory (physical) address
3487                              <1>       ; OUTPUT ->
3488                              <1>       ;     EAX = physical page address
3489                              <1>       ;     (EDX = PTE value)
3490                              <1>       ;     EBX = virtual address
3491                              <1>       ;
3492                              <1>       ;     CF = 1 -> error
3493                              <1>       ;
3494                              <1>       ; Modified Registers -> EAX, EDX
3495                              <1>       ;
3496 00005B6E 50                 <1>       push   eax
3497 00005B6F A1[B8030300]        <1>       mov    eax, [u.pgdir] ; 20/07/2015
3498 00005B74 E837000000          <1>       call   get_pde
3499                              <1>              ; EDX = PDE address
3500                              <1>              ; EAX = PDE value
3501 00005B79 5A                  <1>       pop    edx ; physical page address
3502 00005B7A 722A                <1>       jc     short spte_err ; PDE not present
3503                              <1>       ;
3504 00005B7C 53                  <1>       push   ebx ; 24/07/2015
3505 00005B7D 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
3506                              <1>                          ; EDX = PT address (physical)
3507 00005B81 C1EB0C              <1>       shr    ebx, PAGE_SHIFT ; 12
3508 00005B84 81E3FF030000        <1>       and    ebx, PTE_MASK; 03FFh
3509                              <1>                     ; clear higher 10 bits (PD bits)
3510 00005B8A C1E302              <1>       shl    ebx, 2   ; offset to page table (1024*4)
3511 00005B8D 01C3                <1>       add    ebx, eax
3512                              <1>       ;
3513 00005B8F 8B03                <1>       mov    eax, [ebx] ; Old PTE value
3514 00005B91 A801                <1>       test   al, 1
3515 00005B93 740C                <1>       jz     short spte_0
3516 00005B95 09D2                <1>       or     edx, edx
3517 00005B97 750F                <1>       jnz    short spte_1
3518 00005B99 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 bits
3519 00005B9D 89C2                <1>       mov    edx, eax
3520 00005B9F EB09                <1>       jmp    short spte_2
3521                              <1> spte_0:
3522                              <1>       ; If this PTE contains a swap (disk) address,
3523                              <1>       ; it can be updated by using 'swap_in' procedure
3524                              <1>       ; only!
3525 00005BA1 21C0                <1>       and    eax, eax
3526 00005BA3 7403                <1>       jz     short spte_1
3527                              <1>       ; 24/07/2015
3528                              <1>       ; swapped page ! (on disk)
3529 00005BA5 5B                  <1>       pop    ebx
3530                              <1> spte_err:
3531 00005BA6 F9                  <1>       stc
3532 00005BA7 C3                  <1>       retn
3533                              <1> spte_1:
3534 00005BA8 89D0                <1>       mov    eax, edx
3535                              <1> spte_2:
3536 00005BAA 09CA                <1>       or     edx, ecx
3537                              <1>       ; 23/06/2015
3538 00005BAC 8913                <1>       mov    [ebx], edx ; PTE value in EDX
3539                              <1>       ; 24/07/2015
3540 00005BAE 5B                  <1>       pop    ebx
3541 00005BAF C3                  <1>       retn
3542                              <1>
3543                              <1> get_pde:    ; Get present value of the relevant PDE
3544                              <1>       ; 20/07/2015
3545                              <1>       ; 18/04/2015
3546                              <1>       ; 12/04/2015
3547                              <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
3548                              <1>       ;
3549                              <1>       ; INPUT ->
3550                              <1>       ;     EBX = virtual (linear) address
3551                              <1>       ;     EAX = page directory (physical) address
3552                              <1>       ; OUTPUT ->
3553                              <1>       ;     EDX = Page directory entry address
3554                              <1>       ;     EAX = Page directory entry value
3555                              <1>       ;     CF = 1 -> PDE not present or invalid ?
3556                              <1>       ; Modified Registers -> EDX, EAX
3557                              <1>       ;
3558 00005BB0 89DA                <1>       mov    edx, ebx
3559 00005BB2 C1EA16              <1>       shr    edx, PAGE_D_SHIFT ; 22  (12+10)
3560 00005BB5 C1E202              <1>       shl    edx, 2 ; offset to page directory (1024*4)
```

```
3561 00005BB8 01C2              <1>       add   edx, eax ; page directory address (physical)
3562 00005BBA 8B02              <1>       mov   eax, [edx]
3563 00005BBC A801              <1>       test  al, PDE_A_PRESENT ; page table is present or not !
3564 00005BBE 751F              <1>       jnz   short gpte_retn
3565 00005BC0 F9                <1>       stc
3566                            <1> gpde_retn:
3567 00005BC1 C3                <1>       retn
3568                            <1>
3569                            <1> get_pte:
3570                            <1>             ; Get present value of the relevant PTE
3571                            <1>       ; 29/07/2015
3572                            <1>       ; 20/07/2015
3573                            <1>       ; 18/04/2015
3574                            <1>       ; 12/04/2015
3575                            <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
3576                            <1>       ;
3577                            <1>       ; INPUT ->
3578                            <1>       ;     EBX = virtual (linear) address
3579                            <1>       ;     EAX = page directory (physical) address
3580                            <1>       ; OUTPUT ->
3581                            <1>       ;     EDX = Page table entry address (if CF=0)
3582                            <1>       ;           Page directory entry address (if CF=1)
3583                            <1>       ;            (Bit 0 value is 0 if PT is not present)
3584                            <1>       ;     EAX = Page table entry value (page address)
3585                            <1>       ;     CF = 1 -> PDE not present or invalid ?
3586                            <1>       ; Modified Registers -> EAX, EDX
3587                            <1>       ;
3588 00005BC2 E8E9FFFFFF        <1>       call  get_pde
3589 00005BC7 72F8              <1>       jc    short gpde_retn     ; page table is not present
3590                            <1>       ;jnc  short gpte_1
3591                            <1>       ;retn
3592                            <1> ;gpte_1:
3593 00005BC9 662500F0          <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
3594 00005BCD 89DA              <1>       mov   edx, ebx
3595 00005BCF C1EA0C            <1>       shr   edx, PAGE_SHIFT ; 12
3596 00005BD2 81E2FF030000      <1>       and   edx, PTE_MASK; 03FFh
3597                            <1>                  ; clear higher 10 bits (PD bits)
3598 00005BD8 C1E202            <1>       shl   edx, 2 ; offset from start of page table (1024*4)
3599 00005BDB 01C2              <1>       add   edx, eax
3600 00005BDD 8B02              <1>       mov   eax, [edx]
3601                            <1> gpte_retn:
3602 00005BDF C3                <1>       retn
3603                            <1>
3604                            <1> deallocate_page_dir:
3605                            <1>       ; 15/09/2015
3606                            <1>       ; 05/08/2015
3607                            <1>       ; 30/04/2015
3608                            <1>       ; 28/04/2015
3609                            <1>       ; 17/10/2014
3610                            <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
3611                            <1>       ;
3612                            <1>       ; INPUT ->
3613                            <1>       ;     EAX = PHYSICAL ADDRESS OF THE PAGE DIRECTORY (CHILD)
3614                            <1>       ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
3615                            <1>       ; OUTPUT ->
3616                            <1>       ;     All of page tables in the page directory
3617                            <1>       ;     and page dir's itself will be deallocated
3618                            <1>       ;     except 'read only' duplicated pages (will be converted
3619                            <1>       ;     to writable pages).
3620                            <1>       ;
3621                            <1>       ; Modified Registers -> EAX
3622                            <1>       ;
3623                            <1>       ;
3624 00005BE0 56                <1>       push  esi
3625 00005BE1 51                <1>       push  ecx
3626 00005BE2 50                <1>       push  eax
3627 00005BE3 89C6              <1>       mov   esi, eax
3628 00005BE5 31C9              <1>       xor   ecx, ecx
3629                            <1>       ; The 1st PDE points to Kernel Page Table 0 (the 1st 4MB),
3630                            <1>       ; it must not be deallocated
3631 00005BE7 890E              <1>       mov   [esi], ecx ; 0 ; clear PDE 0
3632                            <1> dapd_0:
3633 00005BE9 AD                <1>       lodsd
3634 00005BEA A801              <1>       test  al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
3635 00005BEC 7409              <1>       jz    short dapd_1
3636 00005BEE 662500F0          <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3637 00005BF2 E812000000        <1>       call  deallocate_page_table
3638                            <1> dapd_1:
3639 00005BF7 41                <1>       inc   ecx ; page directory entry index
3640 00005BF8 81F900040000      <1>       cmp   ecx, PAGE_SIZE / 4 ; 1024
3641 00005BFE 72E9              <1>       jb    short dapd_0
3642                            <1> dapd_2:
3643 00005C00 58                <1>       pop   eax
3644 00005C01 E87F000000        <1>       call  deallocate_page     ; deallocate the page dir's itself
3645 00005C06 59                <1>       pop   ecx
3646 00005C07 5E                <1>       pop   esi
3647 00005C08 C3                <1>       retn
3648                            <1>
3649                            <1> deallocate_page_table:
3650                            <1>       ; 12/07/2016
3651                            <1>       ; 19/09/2015
3652                            <1>       ; 15/09/2015
3653                            <1>       ; 05/08/2015
3654                            <1>       ; 30/04/2015
3655                            <1>       ; 28/04/2015
3656                            <1>       ; 24/10/2014
3657                            <1>       ; 23/10/2014
3658                            <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
3659                            <1>       ;
3660                            <1>       ; INPUT ->
3661                            <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE PAGE TABLE
3662                            <1>       ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
3663                            <1>       ;     (ECX = page directory entry index)
3664                            <1>       ; OUTPUT ->
3665                            <1>       ;     All of pages in the page table and page table's itself
```

```
3666                                    <1>       ;     will be deallocated except 'read only' duplicated pages
3667                                    <1>       ;     (will be converted to writable pages).
3668                                    <1>       ;
3669                                    <1>       ; Modified Registers -> EAX
3670                                    <1>       ;
3671 00005C09 56                       <1>       push  esi
3672 00005C0A 57                       <1>       push  edi
3673 00005C0B 52                       <1>       push  edx
3674 00005C0C 50                       <1>       push  eax ; *
3675 00005C0D 89C6                     <1>       mov   esi, eax
3676 00005C0F 31FF                     <1>       xor   edi, edi ; 0
3677                                    <1> dapt_0:
3678 00005C11 AD                       <1>       lodsd
3679 00005C12 A801                     <1>       test  al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
3680 00005C14 7441                     <1>       jz    short dapt_1
3681                                    <1>       ;
3682 00005C16 A802                     <1>       test  al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
3683                                    <1>                               ; (must be 1)
3684 00005C18 754C                     <1>       jnz   short dapt_3
3685                                    <1>       ; Read only -duplicated- page (belongs to a parent or a child)
3686 00005C1A 66A90002                 <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
3687                                    <1>                                  ; as child's page ?
3688 00005C1E 7451                     <1>       jz    short dapt_4 ; Clear PTE but don't deallocate the page!
3689                                    <1>       ; check the parent's PTE value is read only & same page or not..
3690                                    <1>       ; ECX = page directory entry index (0-1023)
3691 00005C20 53                       <1>       push  ebx
3692 00005C21 51                       <1>       push  ecx
3693 00005C22 66C1E102                 <1>       shl   cx, 2 ; *4
3694 00005C26 01CB                     <1>       add   ebx, ecx ; PDE offset (for the parent)
3695 00005C28 8B0B                     <1>       mov   ecx, [ebx]
3696 00005C2A F6C101                   <1>       test  cl, PDE_A_PRESENT ; present (valid) or not ?
3697 00005C2D 7435                     <1>       jz    short dapt_2 ; parent process does not use this page
3698 00005C2F 6681E100F0               <1>       and   cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
3699                                    <1>       ; EDI = page table entry index (0-1023)
3700 00005C34 89FA                     <1>       mov   edx, edi
3701 00005C36 66C1E202                 <1>       shl   dx, 2 ; *4
3702 00005C3A 01CA                     <1>       add   edx, ecx ; PTE offset (for the parent)
3703 00005C3C 8B1A                     <1>       mov   ebx, [edx]
3704 00005C3E F6C301                   <1>       test  bl, PTE_A_PRESENT ; present or not ?
3705 00005C41 7421                     <1>       jz    short dapt_2 ; parent process does not use this page
3706 00005C43 662500F0                 <1>       and   ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3707 00005C47 6681E300F0               <1>       and   bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3708 00005C4C 39D8                     <1>       cmp   eax, ebx     ; parent's and child's pages are same ?
3709 00005C4E 7514                     <1>       jne   short dapt_2 ; not same page
3710                                    <1>                          ; deallocate the child's page
3711 00005C50 800A02                   <1>         or      byte [edx], PTE_A_WRITE ; convert to writable page (parent)
3712 00005C53 59                       <1>       pop   ecx
3713 00005C54 5B                       <1>       pop   ebx
3714 00005C55 EB1A                     <1>       jmp   short dapt_4
3715                                    <1> dapt_1:
3716 00005C57 09C0                     <1>       or    eax, eax     ; swapped page ?
3717 00005C59 741D                     <1>       jz    short dapt_5 ; no
3718                                    <1>                          ; yes
3719 00005C5B D1E8                     <1>       shr   eax, 1
3720 00005C5D E8CA040000               <1>       call  unlink_swap_block ; Deallocate swapped page block
3721                                    <1>                               ; on the swap disk (or in file)
3722 00005C62 EB14                     <1>       jmp   short dapt_5
3723                                    <1> dapt_2:
3724 00005C64 59                       <1>       pop   ecx
3725 00005C65 5B                       <1>       pop   ebx
3726                                    <1> dapt_3:
3727                                    <1>       ; 12/07/2016
3728 00005C66 66A90004                 <1>       test  ax, PTE_SHARED ; shared or direct memory access indicator
3729 00005C6A 7505                     <1>       jnz   short dapt_4   ; AVL bit 1 = 1, do not deallocate this page!
3730                                    <1>       ;
3731                                    <1>       ;and  ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3732 00005C6C E814000000               <1>       call  deallocate_page ; set the mem allocation bit of this page
3733                                    <1> dapt_4:
3734 00005C71 C746FC00000000           <1>       mov   dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
3735                                    <1> dapt_5:
3736 00005C78 47                       <1>       inc   edi ; page table entry index
3737 00005C79 81FF00040000             <1>       cmp   edi, PAGE_SIZE / 4 ; 1024
3738 00005C7F 7290                     <1>       jb    short dapt_0
3739                                    <1>       ;
3740 00005C81 58                       <1>       pop   eax ; *
3741 00005C82 5A                       <1>       pop   edx
3742 00005C83 5F                       <1>       pop   edi
3743 00005C84 5E                       <1>       pop   esi
3744                                    <1>       ;
3745                                    <1>       ;call  deallocate_page     ; deallocate the page table's itself
3746                                    <1>       ;retn
3747                                    <1>
3748                                    <1> deallocate_page:
3749                                    <1>       ; 15/09/2015
3750                                    <1>       ; 28/04/2015
3751                                    <1>       ; 10/03/2015
3752                                    <1>       ; 17/10/2014
3753                                    <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
3754                                    <1>       ;
3755                                    <1>       ; INPUT ->
3756                                    <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
3757                                    <1>       ; OUTPUT ->
3758                                    <1>       ;     [free_pages] is increased
3759                                    <1>       ;     (corresponding MEMORY ALLOCATION TABLE bit is SET)
3760                                    <1>       ;     CF = 1 if the page is already deallocated
3761                                    <1>       ;            (or not allocated) before.
3762                                    <1>       ;
3763                                    <1>       ; Modified Registers -> EAX
3764                                    <1>       ;
3765 00005C85 53                       <1>       push  ebx
3766 00005C86 52                       <1>       push  edx
3767                                    <1>       ;
3768 00005C87 C1E80C                   <1>       shr   eax, PAGE_SHIFT      ; shift physical address to
3769                                    <1>                                ; 12 bits right
3770                                    <1>                                ; to get page number
```

```
00005C8A 89C2                 <1>       mov    edx, eax
                              <1>       ; 15/09/2015
00005C8C C1EA03               <1>       shr    edx, 3             ; to get offset to M.A.T.
                              <1>                                 ; (1 allocation bit = 1 page)
                              <1>                                 ; (1 allocation bytes = 8 pages)
00005C8F 80E2FC               <1>       and    dl, 0FCh           ; clear lower 2 bits
                              <1>                                 ; (to get 32 bit position)
                              <1>       ;
00005C92 BB00001000           <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address
00005C97 01D3                 <1>       add    ebx, edx
00005C99 83E01F               <1>       and    eax, 1Fh           ; lower 5 bits only
                              <1>                                 ; (allocation bit position)
00005C9C 3B15[FC880100]       <1>       cmp    edx, [next_page]     ; is the new free page address lower
                              <1>                                 ; than the address in 'next_page' ?
                              <1>                                 ; (next/first free page value)
00005CA2 7306                 <1>       jnb    short dap_1        ; no
00005CA4 8915[FC880100]       <1>       mov    [next_page], edx     ; yes
                              <1> dap_1:
00005CAA 0FAB03               <1>       bts    [ebx], eax         ; unlink/release/deallocate page
                              <1>                                 ; set relevant bit to 1.
                              <1>                                 ; set CF to the previous bit value
                              <1>       ;cmc                      ; complement carry flag
                              <1>       ;jc    short dap_2        ; do not increase free_pages count
                              <1>                                 ; if the page is already deallocated
                              <1>                                 ; before.
00005CAD FF05[F8880100]       <1>         inc     dword [free_pages]
                              <1> dap_2:
00005CB3 5A                   <1>       pop    edx
00005CB4 5B                   <1>       pop    ebx
00005CB5 C3                   <1>       retn
                              <1>
                              <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                              <1> ;;                                                              ;;
                              <1> ;; Copyright (C) KolibriOS team 2004-2012. All rights reserved. ;;
                              <1> ;; Distributed under terms of the GNU General Public License    ;;
                              <1> ;;                                                              ;;
                              <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                              <1>
                              <1> ;;$Revision: 5057 $
                              <1>
                              <1>
                              <1> ;;align 4
                              <1> ;;proc alloc_page
                              <1>
                              <1> ;;        pushfd
                              <1> ;;        cli
                              <1> ;;        push    ebx
                              <1> ;;;//-
                              <1> ;;        cmp     [pg_data.pages_free], 1
                              <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>       ;
00005CB6 E8ECFDFFFF             <1>       call   allocate_page
00005CBB 723E                   <1>       jc     short dpd_err
                                <1>       ;
00005CBD 55                     <1>       push   ebp ; 20/07/2015
00005CBE 56                     <1>       push   esi
00005CBF 57                     <1>       push   edi
00005CC0 53                     <1>       push   ebx
00005CC1 51                     <1>       push   ecx
00005CC2 8B35[B8030300]         <1>       mov    esi, [u.pgdir]
00005CC8 89C7                   <1>       mov    edi, eax
00005CCA 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)
00005CCB A5                     <1>       movsd
00005CCC BD00004000             <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
00005CD1 B9FF030000             <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
                                <1> dpd_0:
00005CD6 AD                     <1>       lodsd
                                <1>       ;or    eax, eax
                                <1>         ;jnz     short dpd_1
00005CD7 A801                   <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
00005CD9 7508                   <1>       jnz    short dpd_1
                                <1>       ; 20/07/2015 (virtual address at the end of the page table)
00005CDB 81C500004000           <1>       add    ebp, 1024*4096 ; page size * PTE count
00005CE1 EB0F                   <1>       jmp    short dpd_2
                                <1> dpd_1:
00005CE3 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
00005CE7 89C3                   <1>       mov    ebx, eax
                                <1>       ; EBX = Parent's page table address
00005CE9 E81F000000             <1>       call   duplicate_page_table
00005CEE 720C                   <1>       jc     short dpd_p_err
                                <1>       ; EAX = Child's page table address
00005CF0 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:
00005CF2 AB                     <1>       stosd
00005CF3 E2E1                   <1>       loop   dpd_0
                                <1>       ;
00005CF5 58                     <1>       pop    eax  ; restore child's page directory address
                                <1> dpd_3:
00005CF6 59                     <1>       pop    ecx
00005CF7 5B                     <1>       pop    ebx
00005CF8 5F                     <1>       pop    edi
00005CF9 5E                     <1>       pop    esi
00005CFA 5D                     <1>       pop    ebp ; 20/07/2015
                                <1> dpd_err:
00005CFB C3                     <1>       retn
                                <1> dpd_p_err:
                                <1>       ; release the allocated pages missing (recover free space)
00005CFC 58                     <1>       pop    eax  ; the new page directory address (physical)
00005CFD 8B1D[B8030300]         <1>       mov    ebx, [u.pgdir] ; parent's page directory address
00005D03 E8D8FEFFFF             <1>       call   deallocate_page_dir
00005D08 29C0                   <1>       sub    eax, eax ; 0
00005D0A F9                     <1>       stc
00005D0B EBE9                   <1>       jmp    short dpd_3
                                <1>
                                <1> duplicate_page_table:
                                <1>       ; 20/02/2017
                                <1>       ; 21/09/2015
                                <1>       ; 20/07/2015
                                <1>       ; 05/05/2015
                                <1>       ; 28/04/2015
                                <1>       ; 27/04/2015
                                <1>       ; 18/04/2015
                                <1>       ; 18/10/2014
                                <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
                                <1>       ;       20/02/2017
                                <1>       ;      EBP = Linear address of the page (from 'duplicate_page_dir')
                                <1>       ;            (Linear address = CORE + user's virtual address)
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
                                <1>       ;            (with 'read only' attribute of page table entries)
                                <1>       ;      20/02/2017
                                <1>       ;      EBP = Next linear page address (for 'duplicate_page_dir')
                                <1>       ;
                                <1>       ;      CF = 1 -> error
                                <1>       ;
                                <1>       ; Modified Registers -> EBP (except EAX)
                                <1>       ;
00005D0D E895FDFFFF             <1>       call   allocate_page
00005D12 726A                   <1>       jc     short dpt_err
                                <1>       ;
00005D14 50                     <1>       push   eax ; *
00005D15 56                     <1>       push   esi
00005D16 57                     <1>       push   edi
00005D17 52                     <1>       push   edx
00005D18 51                     <1>       push   ecx
                                <1>       ;
00005D19 89DE                   <1>       mov    esi, ebx
00005D1B 89C7                   <1>       mov    edi, eax
00005D1D 89C2                   <1>       mov    edx, eax
00005D1F 81C200100000           <1>       add    edx, PAGE_SIZE
                                <1> dpt_0:
```

```
3981 00005D25 AD                <1>       lodsd
3982 00005D26 21C0              <1>       and    eax, eax
3983 00005D28 7444              <1>       jz     short dpt_3
3984 00005D2A A801              <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
3985 00005D2C 7507              <1>       jnz    short dpt_1
3986                            <1>       ; 20/07/2015
3987                            <1>       ; ebp = virtual (linear) address of the memory page
3988 00005D2E E83F050000        <1>       call   reload_page ; 28/04/2015
3989 00005D33 7244              <1>       jc     short dpt_p_err
3990                            <1> dpt_1:
3991                            <1>       ; 21/09/2015
3992 00005D35 89C1              <1>       mov    ecx, eax
3993 00005D37 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
3994 00005D3B F6C102            <1>       test   cl, PTE_A_WRITE ; writable page ?
3995 00005D3E 7525              <1>       jnz    short dpt_2
3996                            <1>       ; Read only (parent) page
3997                            <1>       ;      - there is a third process which uses this page -
3998                            <1>       ; Allocate a new page for the child process
3999 00005D40 E862FDFFFF        <1>       call   allocate_page
4000 00005D45 7232              <1>       jc     short dpt_p_err
4001 00005D47 57                <1>       push   edi
4002 00005D48 56                <1>       push   esi
4003 00005D49 89CE              <1>       mov    esi, ecx
4004 00005D4B 89C7              <1>       mov    edi, eax
4005 00005D4D B900040000        <1>       mov    ecx, PAGE_SIZE/4
4006 00005D52 F3A5              <1>       rep    movsd  ; copy page (4096 bytes)
4007 00005D54 5E                <1>       pop    esi
4008 00005D55 5F                <1>       pop    edi
4009                            <1>       ;
4010 00005D56 53                <1>       push   ebx
4011 00005D57 50                <1>       push   eax
4012                            <1>       ; 20/07/2015
4013 00005D58 89EB              <1>       mov    ebx, ebp
4014                            <1>       ; ebx = virtual (linear) address of the memory page
4015 00005D5A E887030000        <1>       call   add_to_swap_queue
4016 00005D5F 58                <1>       pop    eax
4017 00005D60 5B                <1>       pop    ebx
4018                            <1>       ; 21/09/2015
4019 00005D61 0C07              <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
4020                            <1>              ; user + writable + present page
4021 00005D63 EB09              <1>       jmp    short dpt_3
4022                            <1> dpt_2:
4023                            <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
4024 00005D65 0C05              <1>       or     al, PTE_A_USER+PTE_A_PRESENT
4025                            <1>                  ; (read only page!)
4026 00005D67 8946FC            <1>       mov    [esi-4], eax ; update parent's PTE
4027 00005D6A 660D0002          <1>       or      ax, PTE_DUPLICATED  ; (read only page & duplicated PTE!)
4028                            <1> dpt_3:
4029 00005D6E AB                <1>       stosd  ; EDI points to child's PTE
4030                            <1>       ;
4031 00005D6F 81C500100000      <1>       add    ebp, 4096 ; 20/07/2015 (next page)
4032                            <1>       ;
4033 00005D75 39D7              <1>       cmp    edi, edx
4034 00005D77 72AC              <1>       jb     short dpt_0
4035                            <1> dpt_p_err:
4036 00005D79 59                <1>       pop    ecx
4037 00005D7A 5A                <1>       pop    edx
4038 00005D7B 5F                <1>       pop    edi
4039 00005D7C 5E                <1>       pop    esi
4040 00005D7D 58                <1>       pop    eax ; *
4041                            <1> dpt_err:
4042 00005D7E C3                <1>       retn
4043                            <1>
4044                            <1> page_fault_handler:      ; CPU EXCEPTION 0Eh (14) : Page Fault !
4045                            <1>       ; 21/09/2015
4046                            <1>       ; 19/09/2015
4047                            <1>       ; 17/09/2015
4048                            <1>       ; 28/08/2015
4049                            <1>       ; 20/07/2015
4050                            <1>       ; 28/06/2015
4051                            <1>       ; 03/05/2015
4052                            <1>       ; 30/04/2015
4053                            <1>       ; 18/04/2015
4054                            <1>       ; 12/04/2015
4055                            <1>       ; 30/10/2014
4056                            <1>       ; 11/09/2014
4057                            <1>       ; 10/09/2014 (Retro UNIX 386 v1 - beginning)
4058                            <1>       ;
4059                            <1>       ; Note: This is not an interrupt/exception handler.
4060                            <1>       ;     This is a 'page fault remedy' subroutine
4061                            <1>       ;     which will be called by standard/uniform
4062                            <1>       ;     exception handler.
4063                            <1>       ;
4064                            <1>       ; INPUT ->
4065                            <1>       ;     [error_code] = 32 bit ERROR CODE (lower 5 bits are valid)
4066                            <1>       ;
4067                            <1>       ;     cr2 = the virtual (linear) address
4068                            <1>       ;           which has caused to page fault (19/09/2015)
4069                            <1>       ;
4070                            <1>       ; OUTPUT ->
4071                            <1>       ;     (corresponding PAGE TABLE ENTRY is mapped/set)
4072                            <1>       ;     EAX = 0 -> no error
4073                            <1>       ;     EAX > 0 -> error code in EAX (also CF = 1)
4074                            <1>       ;
4075                            <1>       ; Modified Registers -> none (except EAX)
4076                            <1>       ;
4077                            <1>         ;
4078                            <1>         ; ERROR CODE:
4079                            <1>       ;      31  .....   4   3  2   1  0
4080                            <1>       ;     +---+-- --+---+---+---+---+---+---+
4081                            <1>       ;     |   Reserved  | I | R | U | W | P |
4082                            <1>       ;     +---+-- --+---+---+---+---+---+---+
4083                            <1>       ;
4084                            <1>       ; P : PRESENT -     When set, the page fault was caused by
4085                            <1>       ;             a page-protection violation. When not set,
```

```
                        <1>     ;            it was caused by a non-present page.
                        <1>     ; W : WRITE   -     When set, the page fault was caused by
                        <1>     ;            a page write. When not set, it was caused
                        <1>     ;            by a page read.
                        <1>     ; U : USER    -     When set, the page fault was caused
                        <1>     ;            while CPL = 3.
                        <1>     ;            This does not necessarily mean that
                        <1>     ;            the page fault was a privilege violation.
                        <1>     ; R : RESERVD -     When set, the page fault was caused by
                        <1>     ;     WRITE  reading a 1 in a reserved field.
                        <1>     ; I : INSTRUC -     When set, the page fault was caused by
                        <1>     ;     FETCH  an instruction fetch
                        <1>     ;
                        <1>     ;; x86 (32 bit) VIRTUAL ADDRESS TRANSLATION
                        <1>     ;  31               22                  12 11                   0
                        <1>     ; +-------------------+-------------------+-----------------------+
                        <1>           ; | PAGE DIR. ENTRY # | PAGE TAB. ENTRY # |        OFFSET         |
                        <1>           ; +-------------------+-------------------+-----------------------+
                        <1>     ;
                        <1>
                        <1>     ;; CR3 REGISTER (Control Register 3)
                        <1>     ;  31                                   12             5 4 3 2   0
                        <1>     ; +---------------------------------------+-------------+---+-----+
                        <1>           ; |                                       |          |P|P|     |
                        <1>           ; |   PAGE DIRECTORY TABLE BASE ADDRESS   |  reserved      |C|W|rsvrd|
                        <1>           ; |                                       |          |D|T|     |
                        <1>     ; +---------------------------------------+-------------+---+-----+
                        <1>     ;
                        <1>     ;      PWT    - WRITE THROUGH
                        <1>     ;      PCD    - CACHE DISABLE
                        <1>     ;
                        <1>     ;
                        <1>     ;; x86 PAGE DIRECTORY ENTRY (4 KByte Page)
                        <1>     ;  31                                   12 11  9 8 7 6 5 4 3 2 1 0
                        <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>
00005D7F 53             <1>     push   ebx
00005D80 52             <1>     push   edx
00005D81 51             <1>     push   ecx
                        <1>     ;
                        <1>     ; 21/09/2015 (debugging)
00005D82 FF05[CC030300] <1>     inc    dword [u.pfcount] ; page fault count for running process
00005D88 FF05[80050300] <1>     inc    dword [PF_Count] ; total page fault count
```

```
                                <1>       ; 28/06/2015
                                <1>       ;mov   edx, [error_code] ; Lower 5 bits are valid
00005D8E 8A15[78050300]         <1>       mov    dl, [error_code]
                                <1>       ;
00005D94 F6C201                 <1>       test   dl, 1  ; page fault was caused by a non-present page
                                <1>                     ; sign
00005D97 7422                   <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>       ;
00005D99 F6C202                 <1>       test   dl, 2  ; page fault was caused by a page write
                                <1>                     ; sign
00005D9C 0F84AB000000           <1>         jz      pfh_p_err
                                <1>       ; 31/08/2015
00005DA2 F6C204                 <1>       test   dl, 4  ; page fault was caused while CPL = 3 (user mode)
                                <1>                     ; sign.  (U+W+P = 4+2+1 = 7)
00005DA5 0F84A2000000           <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>       ;
00005DAB 0F20D3                 <1>       mov    ebx, cr2 ; CR2 contains the linear address
                                <1>                       ; which has caused to page fault
00005DAE E8A2000000             <1>       call   copy_page
00005DB3 0F828D000000           <1>         jc      pfh_im_err ; insufficient memory
                                <1>       ;
00005DB9 EB7D                   <1>         jmp     pfh_cpp_ok
                                <1>       ;
                                <1> pfh_alloc_np:
00005DBB E8E7FCFFFF             <1>       call   allocate_page; (allocate a new page)
00005DC0 0F8280000000           <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)
00005DC6 80E204                 <1>       and    dl, 4  ; CPL = 3 ?
00005DC9 7505                   <1>       jnz    short pfh_um
                                <1>                     ; Page fault handler for kernel/system mode (CPL=0)
00005DCB 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)
00005DCE EB06                   <1>       jmp    short pfh_get_pde
                                <1>       ;
                                <1> pfh_um:                     ; Page fault handler for user/appl. mode (CPL=3)
00005DD0 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:
00005DD6 80CA03                 <1>       or     dl, 3  ; USER + WRITE + PRESENT or SYSTEM + WRITE + PRESENT
00005DD9 0F20D1                 <1>       mov    ecx, cr2 ; CR2 contains the virtual address
                                <1>                       ; which has been caused to page fault
                                <1>                       ;
00005DDC C1E914                 <1>       shr    ecx, 20       ; shift 20 bits right
00005DDF 80E1FC                 <1>       and    cl, 0FCh ; mask lower 2 bits to get PDE offset
                                <1>       ;
00005DE2 01CB                   <1>       add    ebx, ecx ; now, EBX points to the relevant page dir entry
00005DE4 8B0B                   <1>       mov    ecx, [ebx] ; physical (base) address of the page table
00005DE6 F6C101                 <1>       test   cl, 1   ; check bit 0 is set (1) or not (0).
00005DE9 740B                   <1>       jz     short pfh_set_pde ; Page directory entry is not valid,
                                <1>                                ; set/validate page directory entry
00005DEB 6681E100F0             <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
00005DF0 89CB                   <1>       mov    ebx, ecx ; Physical address of the page table
00005DF2 89C1                   <1>       mov    ecx, eax ; new page address (physical)
00005DF4 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>       ;
00005DF6 08D0                   <1>       or     al, dl  ; lower 3 bits are used as U/S, R/W, P flags
00005DF8 8903                   <1>       mov    [ebx], eax ; Let's put the new page directory entry here !
00005DFA 30C0                   <1>       xor    al, al  ; clear lower (3..8) bits
00005DFC 89C3                   <1>       mov    ebx, eax
00005DFE E8A4FCFFFF             <1>       call   allocate_page ; (allocate a new page)
00005E03 7241                   <1>       jc     short pfh_im_err   ; 'insufficient memory' error
                                <1> pfh_spde_1:
                                <1>       ; EAX = Physical (base) address of the allocated (new) page
00005E05 89C1                   <1>       mov    ecx, eax
00005E07 E815FDFFFF             <1>       call   clear_page ; Clear page content
                                <1> pfh_get_pte:
00005E0C 0F20D0                 <1>       mov    eax, cr2 ; virtual address
                                <1>                       ; which has been caused to page fault
00005E0F 89C7                   <1>       mov    edi, eax ; 20/07/2015
00005E11 C1E80C                 <1>       shr    eax, 12       ; shift 12 bit right to get
                                <1>                       ; higher 20 bits of the page fault address
00005E14 25FF030000             <1>       and    eax, 3FFh ; mask PDE# bits, the result is PTE# (0 to 1023)
00005E19 C1E002                 <1>       shl    eax, 2 ; shift 2 bits left to get PTE offset
```

```
4296 00005E1C 01C3              <1>       add    ebx, eax ; now, EBX points to the relevant page table entry
4297 00005E1E 8B03              <1>       mov    eax, [ebx] ; get previous value of pte
4298                            <1>              ; bit 0 of EAX is always 0 (otherwise we would not be here)
4299 00005E20 21C0              <1>       and    eax, eax
4300 00005E22 7410              <1>       jz     short pfh_gpte_1
4301                            <1>       ; 20/07/2015
4302 00005E24 87D9              <1>       xchg   ebx, ecx ; new page address (physical)
4303 00005E26 55                <1>       push   ebp ; 20/07/2015
4304 00005E27 0F20D5            <1>       mov    ebp, cr2
4305                            <1>              ; ECX = physical address of the page table entry
4306                            <1>              ; EBX = Memory page address (physical!)
4307                            <1>              ; EAX = Swap disk (offset) address
4308                            <1>              ; EBP = virtual address (page fault address)
4309 00005E2A E8B7000000        <1>       call   swap_in
4310 00005E2F 5D                <1>       pop    ebp
4311 00005E30 7210              <1>       jc      short pfh_err_retn
4312 00005E32 87CB              <1>       xchg   ecx, ebx
4313                            <1>              ; EBX = physical address of the page table entry
4314                            <1>              ; ECX = new page
4315                            <1> pfh_gpte_1:
4316 00005E34 08D1              <1>       or     cl, dl ; lower 3 bits are used as U/S, R/W, P flags
4317 00005E36 890B              <1>       mov    [ebx], ecx ; Let's put the new page table entry here !
4318                            <1> pfh_cpp_ok:
4319                            <1>       ; 20/07/2015
4320 00005E38 0F20D3            <1>       mov    ebx, cr2
4321 00005E3B E8A6020000        <1>       call   add_to_swap_queue
4322                            <1>       ;
4323                            <1>       ; The new PTE (which contains the new page) will be added to
4324                            <1>       ; the swap queue, here.
4325                            <1>       ; (Later, if memory will become insufficient,
4326                            <1>       ; one page will be swapped out which is at the head of
4327                            <1>       ; the swap queue by using FIFO and access check methods.)
4328                            <1>       ;
4329 00005E40 31C0              <1>       xor    eax, eax  ; 0
4330                            <1>       ;
4331                            <1> pfh_err_retn:
4332 00005E42 59                <1>       pop    ecx
4333 00005E43 5A                <1>       pop    edx
4334 00005E44 5B                <1>       pop    ebx
4335 00005E45 C3                <1>       retn
4336                            <1>
4337                            <1> pfh_im_err:
4338 00005E46 B8E4000000        <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_IM ; Error code in AX
4339                            <1>                     ; Major (Primary) Error: Page Fault
4340                            <1>                     ; Minor (Secondary) Error: Insufficient Memory !
4341 00005E4B EBF5              <1>       jmp    short pfh_err_retn
4342                            <1>
4343                            <1>
4344                            <1> pfh_p_err: ; 09/03/2015
4345                            <1> pfh_pv_err:
4346                            <1>       ; Page fault was caused by a protection-violation
4347 00005E4D B8E6000000        <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_PV ; Error code in AX
4348                            <1>                     ; Major (Primary) Error: Page Fault
4349                            <1>                     ; Minor (Secondary) Error: Protection violation !
4350 00005E52 F9                <1>       stc
4351 00005E53 EBED              <1>       jmp    short pfh_err_retn
4352                            <1>
4353                            <1> copy_page:
4354                            <1>       ; 22/09/2015
4355                            <1>       ; 21/09/2015
4356                            <1>       ; 19/09/2015
4357                            <1>       ; 07/09/2015
4358                            <1>       ; 31/08/2015
4359                            <1>       ; 20/07/2015
4360                            <1>       ; 05/05/2015
4361                            <1>       ; 03/05/2015
4362                            <1>       ; 18/04/2015
4363                            <1>       ; 12/04/2015
4364                            <1>       ; 30/10/2014
4365                            <1>       ; 18/10/2014 (Retro UNIX 386 v1 - beginning)
4366                            <1>       ;
4367                            <1>       ; INPUT ->
4368                            <1>       ;      EBX = Virtual (linear) address of source page
4369                            <1>       ;           (Page fault address)
4370                            <1>       ; OUTPUT ->
4371                            <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
4372                            <1>       ;      (corresponding PAGE TABLE ENTRY is mapped/set)
4373                            <1>       ;      EAX = 0 (CF = 1)
4374                            <1>       ;            if there is not a free page to be allocated
4375                            <1>       ;      (page content of the source page will be copied
4376                            <1>       ;      onto the target/new page)
4377                            <1>       ;
4378                            <1>       ; Modified Registers -> ecx, ebx (except EAX)
4379                            <1>       ;
4380 00005E55 56                <1>       push   esi
4381 00005E56 57                <1>       push   edi
4382                            <1>       ;push  ebx
4383                            <1>       ;push  ecx
4384 00005E57 31F6              <1>       xor    esi, esi
4385 00005E59 C1EB0C            <1>       shr    ebx, 12 ; shift 12 bits right to get PDE & PTE numbers
4386 00005E5C 89D9              <1>       mov    ecx, ebx ; save page fault address (as 12 bit shifted)
4387 00005E5E C1EB08            <1>       shr    ebx, 8  ; shift 8 bits right and then
4388 00005E61 80E3FC            <1>       and    bl, 0FCh ; mask lower 2 bits to get PDE offset
4389 00005E64 89DF              <1>       mov    edi, ebx ; save it for the parent of current process
4390 00005E66 031D[B8030300]    <1>       add    ebx, [u.pgdir] ; EBX points to the relevant page dir entry
4391 00005E6C 8B03              <1>       mov    eax, [ebx] ; physical (base) address of the page table
4392 00005E6E 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
4393 00005E72 89CB              <1>       mov    ebx, ecx   ; (restore higher 20 bits of page fault address)
4394 00005E74 81E3FF030000      <1>       and    ebx, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
4395 00005E7A 66C1E302          <1>       shl    bx, 2    ; shift 2 bits left to get PTE offset
4396 00005E7E 01C3              <1>       add    ebx, eax   ; EBX points to the relevant page table entry
4397                            <1>       ; 07/09/2015
4398 00005E80 66F7030002        <1>         test    word [ebx], PTE_DUPLICATED ; (Does current process share this
4399                            <1>                                ; read only page as a child process?)
4400 00005E85 7509              <1>       jnz    short cpp_0 ; yes
```

```
4401 00005E87 8B0B              <1>       mov   ecx, [ebx] ; PTE value
4402 00005E89 6681E100F0        <1>       and   cx, PTE_A_CLEAR ; 0F000h  ; clear page attributes
4403 00005E8E EB32              <1>       jmp   short cpp_1
4404                            <1> cpp_0:
4405 00005E90 89FE              <1>       mov   esi, edi
4406 00005E92 0335[BC030300]    <1>       add   esi, [u.ppgdir] ; the parent's page directory entry
4407 00005E98 8B06              <1>       mov   eax, [esi] ; physical (base) address of the page table
4408 00005E9A 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
4409 00005E9E 89CE              <1>       mov   esi, ecx   ; (restore higher 20 bits of page fault address)
4410 00005EA0 81E6FF030000      <1>       and   esi, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
4411 00005EA6 66C1E602          <1>       shl   si, 2    ; shift 2 bits left to get PTE offset
4412 00005EAA 01C6              <1>       add   esi, eax   ; EDX points to the relevant page table entry
4413 00005EAC 8B0E              <1>       mov   ecx, [esi] ; PTE value of the parent process
4414                            <1>       ; 21/09/2015
4415 00005EAE 8B03              <1>       mov   eax, [ebx] ; PTE value of the child process
4416 00005EB0 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear page attributes
4417                            <1>       ;
4418 00005EB4 F6C101            <1>       test  cl, PTE_A_PRESENT ; is it a present/valid page ?
4419 00005EB7 7424              <1>       jz    short cpp_3 ; the parent's page is not same page
4420                            <1>       ;
4421 00005EB9 6681E100F0        <1>       and   cx, PTE_A_CLEAR ; 0F000h ; clear page attributes
4422 00005EBE 39C8              <1>       cmp   eax, ecx   ; Same page?
4423 00005EC0 751B              <1>       jne   short cpp_3 ; Parent page and child page are not same
4424                            <1>                         ; Convert child's page to writable page
4425                            <1> cpp_1:
4426 00005EC2 E8E0FBFFFF        <1>       call  allocate_page
4427 00005EC7 721A              <1>       jc    short cpp_4 ; 'insufficient memory' error
4428 00005EC9 21F6              <1>       and   esi, esi    ; check ESI is valid or not
4429 00005ECB 7405              <1>       jz    short cpp_2
4430                            <1>             ; Convert read only page to writable page
4431                            <1>             ;(for the parent of the current process)
4432                            <1>       ;and  word [esi], PTE_A_CLEAR ; 0F000h
4433                            <1>       ; 22/09/2015
4434 00005ECD 890E              <1>       mov   [esi], ecx
4435 00005ECF 800E07            <1>       or    byte [esi], PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER
4436                            <1>                           ; 1+2+4 = 7
4437                            <1> cpp_2:
4438 00005ED2 89C7              <1>       mov   edi, eax ; new page address of the child process
4439                            <1>       ; 07/09/2015
4440 00005ED4 89CE              <1>       mov   esi, ecx ; the page address of the parent process
4441 00005ED6 B900040000        <1>       mov   ecx, PAGE_SIZE / 4
4442 00005EDB F3A5              <1>       rep   movsd ; 31/08/2015
4443                            <1> cpp_3:
4444 00005EDD 0C07              <1>       or    al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 1+2+4 = 7
4445 00005EDF 8903              <1>       mov   [ebx], eax ; Update PTE
4446 00005EE1 28C0              <1>       sub   al, al ; clear attributes
4447                            <1> cpp_4:
4448                            <1>       ;pop  ecx
4449                            <1>       ;pop  ebx
4450 00005EE3 5F                <1>       pop   edi
4451 00005EE4 5E                <1>       pop   esi
4452 00005EE5 C3                <1>       retn
4453                            <1>
4454                            <1> ;; 28/04/2015
4455                            <1> ;; 24/10/2014
4456                            <1> ;; 21/10/2014 (Retro UNIX 386 v1 - beginning)
4457                            <1> ;; SWAP_PAGE_QUEUE (4096 bytes)
4458                            <1> ;;
4459                            <1> ;;   0000   0001   0002   0003   ....   1020   1021   1022   1023
4460                            <1> ;; +------+------+------+------+-    -+------+------+------+------+
4461                            <1> ;; |  pg1 |  pg2 |  pg3 |  pg4 | .... |pg1021|pg1022|pg1023|pg1024|
4462                            <1> ;; +------+------+------+------+-    -+------+------+------+------+
4463                            <1> ;;
4464                            <1> ;; [swpq_last] = 0 to 4096 (step 4) -> the last position on the queue
4465                            <1> ;;
4466                            <1> ;; Method:
4467                            <1> ;;    Swap page queue is a list of allocated pages with physical
4468                            <1> ;;    addresses (system mode virtual adresses = physical addresses).
4469                            <1> ;;    It is used for 'swap_in' and 'swap_out' procedures.
4470                            <1> ;;    When a new page is being allocated, swap queue is updated
4471                            <1> ;;    by 'swap_queue_shift' procedure, header of the queue (offset 0)
4472                            <1> ;;    is checked for 'accessed' flag. If the 1st page on the queue
4473                            <1> ;;    is 'accessed' or 'read only', it is dropped from the list;
4474                            <1> ;;    other pages from the 2nd to the last (in [swpq_last]) shifted
4475                            <1> ;;    to head then the 2nd page becomes the 1st and '[swpq_last]'
4476                            <1> ;;    offset value becomes it's previous offset value - 4.
4477                            <1> ;;    If the 1st page of the swap page queue is not 'accessed'
4478                            <1> ;;    the queue/list is not shifted.
4479                            <1> ;;    After the queue/list shift, newly allocated page is added
4480                            <1> ;;    to the tail of the queue at the [swpq_count*4] position.
4481                            <1> ;;    But, if [swpq_count] > 1023, the newly allocated page
4482                            <1> ;;    will not be added to the tail of swap page queue.
4483                            <1> ;;
4484                            <1> ;;    During 'swap_out' procedure, swap page queue is checked for
4485                            <1> ;;    the first non-accessed, writable page in the list,
4486                            <1> ;;    from the head to the tail. The list is shifted to left
4487                            <1> ;;    (to the head) till a non-accessed page will be found in the list.
4488                            <1> ;;    Then, this page     is swapped out (to disk) and then it is dropped
4489                            <1> ;;    from the list by a final swap queue shift. [swpq_count] value
4490                            <1> ;;    is changed. If all pages on the queue' are 'accessed',
4491                            <1> ;;    'insufficient memory' error will be returned ('swap_out'
4492                            <1> ;;    procedure will be failed)...
4493                            <1> ;;
4494                            <1> ;;    Note: If the 1st page of the queue is an 'accessed' page,
4495                            <1> ;;    'accessed' flag of the page will be reset (0) and that page
4496                            <1> ;;    (PTE) will be added to the tail of the queue after
4497                            <1> ;;    the check, if [swpq_count] < 1023. If [swpq_count] = 1024
4498                            <1> ;;    the queue will be rotated and the PTE in the head will be
4499                            <1> ;;    added to the tail after resetting 'accessed' bit.
4500                            <1> ;;
4501                            <1> ;;
4502                            <1> ;;
4503                            <1> ;; SWAP DISK/FILE (with 4096 bytes swapped page blocks)
4504                            <1> ;;
4505                            <1> ;;  00000000  00000004  00000008  0000000C   ...   size-8    size-4
```

```
<1> ;; +---------+---------+---------+---------+-- --+---------+---------+
<1> ;; |descriptr| page(1) | page(2) | page(3) | ... |page(n-1)| page(n) |
<1> ;; +---------+---------+---------+---------+-- --+---------+---------+
<1> ;;
<1> ;; [swpd_next] = the first free block address in swapped page records
<1> ;;                  for next free block search by 'swap_out' procedure.
<1> ;; [swpd_size] = swap disk/file size in sectors (512 bytes)
<1> ;;           NOTE: max. possible swap disk size is 1024 GB
<1> ;;           (entire swap space must be accessed by using
<1> ;;           31 bit offset address)
<1> ;; [swpd_free] = free block (4096 bytes) count in swap disk/file space
<1> ;; [swpd_start] = absolute/start address of the swap disk/file
<1> ;;            0 for file, or beginning sector of the swap partition
<1> ;; [swp_drv] = logical drive description table addr. of swap disk/file
<1> ;;
<1> ;;
<1> ;; Method:
<1> ;;    When the memory (ram) becomes insufficient, page allocation
<1> ;;    procedure swaps out a page from memory to the swap disk
<1> ;;    (partition) or swap file to get a new free page at the memory.
<1> ;;    Swapping out is performed by using swap page queue.
<1> ;;
<1> ;;    Allocation block size of swap disk/file is equal to page size
<1> ;;    (4096 bytes). Swapping address (in sectors) is recorded
<1> ;;    into relevant page file entry as 31 bit physical (logical)
<1> ;;    offset address as 1 bit shifted to left for present flag (0).
<1> ;;    Swapped page address is between 1 and swap disk/file size - 4.
<1> ;;    Absolute physical (logical) address of the swapped page is
<1> ;;    calculated by adding offset value to the swap partition's
<1> ;;    start address. If the swap device (disk) is a virtual disk
<1> ;;    or it is a file, start address of the swap disk/volume is 0,
<1> ;;    and offset value is equal to absolute (physical or logical)
<1> ;;    address/position. (It has not to be ZERO if the swap partition
<1> ;;    is in a partitioned virtual hard disk.)
<1> ;;
<1> ;;    Note: Swap addresses are always specified/declared in sectors,
<1> ;;    not in bytes or     in blocks/zones/clusters (4096 bytes) as unit.
<1> ;;
<1> ;;    Swap disk/file allocation is mapped via 'Swap Allocation Table'
<1> ;;    at memory as similar to 'Memory Allocation Table'.
<1> ;;
<1> ;;    Every bit of Swap Allocation Table repsesents one swap block
<1> ;;    (equal to page size) respectively. Bit 0 of the S.A.T. byte 0
<1> ;;    is reserved for swap disk/file block 0 as descriptor block
<1> ;;    (also for compatibility with PTE). If bit value is ZERO,
<1> ;;    it means relevant (respective) block is in use, and,
<1> ;;    of course, if bit value is 1, it means relevant (respective)
<1> ;;      swap disk/file block is free.
<1> ;;    For example: bit 1 of the byte 128 repsesents block 1025
<1> ;;    (128*8+1) or sector (offset) 8200 on the swap disk or
<1> ;;    byte (offset/position) 4198400 in the swap file.
<1> ;;    4GB swap space is represented via 128KB Swap Allocation Table.
<1> ;;    Initial layout of Swap Allocation Table is as follows:
<1> ;;    ------------------------------------------------------------
<1> ;;    01111111111111111111111 .... 11111111111111111111111111111
<1> ;;    ------------------------------------------------------------
<1> ;;    (0 is reserved block, 1s represent free blocks respectively.)
<1> ;;    (Note: Allocation cell/unit of the table is bit, not byte)
<1> ;;
<1> ;;    ............................................................
<1> ;;
<1> ;;    'swap_out' procedure checks 'free_swap_blocks' count at first,
<1> ;;    then it searches Swap Allocation Table if free count is not
<1> ;;    zero. From begining the [swpd_next] dword value, the first bit
<1> ;;    position with value of 1 on the table is converted to swap
<1> ;;    disk/file offset address, in sectors (not 4096 bytes block).
<1> ;;    'ldrv_write' procedure is called with ldrv (logical drive
<1> ;;    number of physical swap disk or virtual swap disk)
<1> ;;    number, sector offset (not absolute sector -LBA- number),
<1> ;;    and sector count (8, 512*8 = 4096) and buffer adress
<1> ;;    (memory page). That will be a direct disk write procedure.
<1> ;;    (for preventing late memory allocation, significant waiting).
<1> ;;    If disk write procedure returns with error or free count of
<1> ;;    swap blocks is ZERO, 'swap_out' procedure will return with
<1> ;;    'insufficient memory error' (cf=1).
<1> ;;
<1> ;;    (Note: Even if free swap disk/file blocks was not zero,
<1> ;;    any disk write error will not be fixed by 'swap_out' procedure,
<1> ;;    in other words, 'swap_out' will not check the table for other
<1> ;;    free blocks after a disk write error. It will return to
<1> ;;    the caller with error (CF=1) which means swapping is failed.
<1> ;;
<1> ;;    After writing the page on to swap disk/file address/sector,
<1> ;;    'swap_out' procesure returns with that swap (offset) sector
<1> ;;    address (cf=0).
<1> ;;
<1> ;;    ............................................................
<1> ;;
<1> ;;    'swap_in' procedure loads addressed (relevant) swap disk or
<1> ;;    file sectors at specified memory page. Then page allocation
<1> ;;    procedure updates relevant page table entry with 'present'
<1> ;;    attribute. If swap disk or file reading fails there is nothing
<1> ;;    to do, except to terminate the process which is the owner of
<1> ;;    the swapped page.
<1> ;;
<1> ;;    'swap_in' procedure sets the relevant/respective bit value
<1> ;;    in the Swap Allocation Table (as free block). 'swap_in' also
<1> ;;    updates [swpd_first] pointer if it is required.
<1> ;;
<1> ;;    ............................................................
<1> ;;
<1> ;;    Note: If [swap_enabled] value is ZERO, that means there is not
<1> ;;    a swap disk or swap file in use... 'swap_in' and 'swap_out'
<1> ;;    procedures ans 'swap page que' procedures will not be active...
<1> ;;    'Insufficient memory' error will be returned by 'swap_out'
```

```
                        <1> ;;    and 'general protection fault' will be returned by 'swap_in'
                        <1> ;;    procedure, if it is called mistakenly (a wrong value in a PTE).
                        <1> ;;
                        <1>
                        <1> swap_in:
                        <1>       ; 31/08/2015
                        <1>       ; 20/07/2015
                        <1>       ; 28/04/2015
                        <1>       ; 18/04/2015
                        <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;     EBX = PHYSICAL (real/flat) ADDRESS OF THE MEMORY PAGE
                        <1>       ;     EBP = VIRTUAL (LINEAR) ADDRESS (page fault address)
                        <1>       ;     EAX = Offset Address for the swapped page on the
                        <1>       ;           swap disk or in the swap file.
                        <1>       ;
                        <1>       ; OUTPUT ->
                        <1>       ;     EAX = 0 if loading at memory has been successful
                        <1>       ;
                        <1>       ;     CF = 1 -> swap disk reading error (disk/file not present
                        <1>       ;             or sector not present or drive not ready
                        <1>       ;          EAX = Error code
                        <1>       ;          [u.error] = EAX
                        <1>       ;                  = The last error code for the process
                        <1>       ;                    (will be reset after returning to user)
                        <1>       ;
                        <1>       ; Modified Registers -> EAX
                        <1>       ;
                        <1>
00005EE6 833D[62050300]00 <1>       cmp     dword [swp_drv], 0
00005EED 7646             <1>       jna     short swpin_dnp_err
                        <1>
00005EEF 3B05[66050300]   <1>       cmp     eax, [swpd_size]
00005EF5 734A             <1>       jnb     short swpin_snp_err
                        <1>
00005EF7 56               <1>       push    esi
00005EF8 53               <1>       push    ebx
00005EF9 51               <1>       push    ecx
00005EFA 8B35[62050300]   <1>       mov     esi, [swp_drv]
00005F00 B908000000       <1>       mov     ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
                        <1>             ; Note: Even if corresponding physical disk's sector
                        <1>             ; size different than 512 bytes, logical disk sector
                        <1>             ; size is 512 bytes and disk reading procedure
                        <1>             ; will be performed for reading 4096 bytes
                        <1>             ; (2*2048, 8*512).
                        <1>       ; ESI = Logical disk description table address
                        <1>       ; EBX = Memory page (buffer) address (physical!)
                        <1>       ; EAX = Sector adress (offset address, logical sector number)
                        <1>       ; ECX = Sector count ; 8 sectors
00005F05 50               <1>       push    eax
00005F06 E8AF020000       <1>       call    logical_disk_read
00005F0B 58               <1>       pop     eax
00005F0C 730C             <1>       jnc     short swpin_read_ok
                        <1>       ;
00005F0E B828000000       <1>       mov     eax, SWP_DISK_READ_ERR ; drive not ready or read error
00005F13 A3[C8030300]     <1>       mov     [u.error], eax
00005F18 EB17             <1>       jmp     short swpin_retn
                        <1>       ;
                        <1> swpin_read_ok:
                        <1>       ; EAX = Offset address (logical sector number)
00005F1A E80D020000       <1>       call    unlink_swap_block  ; Deallocate swap block
                        <1>       ;
                        <1>       ; EBX = Memory page (buffer) address (physical!)
                        <1>       ; 20/07/2015
00005F1F 89EB             <1>       mov     ebx, ebp ; virtual address (page fault address)
00005F21 6681E300F0       <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
00005F26 8A1D[B3030300]   <1>       mov     bl, [u.uno] ; current process number
                        <1>       ; EBX = Virtual (Linear) address & process number combination
00005F2C E8DB000000       <1>       call    swap_queue_shift
                        <1>       ; eax = 0 ; 10/06/2016 (if ebx input > 0, eax output = 0)
                        <1>       ;sub     eax, eax  ; 0 ; Error Code = 0  (no error)
                        <1>       ; zf = 1
                        <1> swpin_retn:
00005F31 59               <1>       pop     ecx
00005F32 5B               <1>       pop     ebx
00005F33 5E               <1>       pop     esi
00005F34 C3               <1>       retn
                        <1>
                        <1> swpin_dnp_err:
00005F35 B829000000       <1>       mov     eax, SWP_DISK_NOT_PRESENT_ERR
                        <1> swpin_err_retn:
00005F3A A3[C8030300]     <1>       mov     [u.error], eax
00005F3F F9               <1>       stc
00005F40 C3               <1>       retn
                        <1>
                        <1> swpin_snp_err:
00005F41 B82A000000       <1>       mov     eax, SWP_SECTOR_NOT_PRESENT_ERR
00005F46 EBF2             <1>       jmp     short swpin_err_retn
                        <1>
                        <1> swap_out:
                        <1>       ; 10/06/2016
                        <1>       ; 07/06/2016
                        <1>         ; 23/05/2016
                        <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>       ; 24/10/2014 - 31/08/2015 (Retro UNIX 386 v1)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;     none
                        <1>       ;
                        <1>       ; OUTPUT ->
                        <1>       ;     EAX = Physical page address (which is swapped out
                        <1>       ;           for allocating a new page)
                        <1>       ;     CF = 1 -> swap disk writing error (disk/file not present
                        <1>       ;             or sector not present or drive not ready
```

```
4716                                    <1>       ;           EAX = Error code
4717                                    <1>       ;           [u.error] = EAX
4718                                    <1>       ;                   = The last error code for the process
4719                                    <1>       ;                     (will be reset after returning to user)
4720                                    <1>       ;
4721                                    <1>       ; Modified Registers -> none (except EAX)
4722                                    <1>       ;
4723 00005F48 66833D[60050300]01       <1>       cmp    word [swpq_count], 1
4724 00005F50 0F82AF000000             <1>         jc      swpout_im_err ; 'insufficient memory'
4725                                    <1>
4726                                    <1>         ;cmp    dword [swp_drv], 1
4727                                    <1>       ;jc    short swpout_dnp_err ; 'swap disk/file not present'
4728                                    <1>
4729 00005F56 833D[6A050300]01         <1>         cmp     dword [swpd_free], 1
4730 00005F5D 0F828F000000             <1>         jc      swpout_nfspc_err ; 'no free space on swap disk'
4731                                    <1>
4732 00005F63 53                       <1>       push   ebx ; *
4733                                    <1> swpout_1:
4734                                    <1>       ; 10/06/2016
4735 00005F64 31DB                     <1>       xor    ebx, ebx ; shift the queue and return a PTE value
4736 00005F66 E8A1000000               <1>       call   swap_queue_shift
4737 00005F6B 21C0                     <1>       and    eax, eax     ; 0 = empty queue (improper entries)
4738 00005F6D 0F848A000000             <1>         jz      swpout_npts_err        ; There is not any proper PTE
4739                                    <1>                                      ; pointer in the swap queue
4740                                    <1>       ; EAX = PTE value of the page
4741                                    <1>       ; EBX = PTE address of the page
4742 00005F73 662500F0                 <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
4743                                    <1>       ;
4744                                    <1>       ; 07/06/2016
4745                                    <1>       ; 19/05/2016
4746                                    <1>       ; check this page is in timer events or not
4747                                    <1>
4748                                    <1> swpout_timer_page_0:
4749 00005F77 52                       <1>       push   edx ; **
4750                                    <1>
4751                                    <1>       ; 07/06/2016
4752 00005F78 803D[83950100]00         <1>       cmp    byte [timer_events], 0
4753 00005F7F 762F                     <1>       jna    short swpout_2
4754                                    <1>       ;
4755 00005F81 8A15[83950100]           <1>       mov    dl, [timer_events]
4756                                    <1>
4757 00005F87 51                       <1>       push   ecx ; ***
4758 00005F88 53                       <1>       push   ebx ; ****
4759 00005F89 BB[60040300]             <1>       mov    ebx, timer_set ; beginning address of timer event
4760                                    <1>                             ; structures
4761                                    <1> swpout_timer_page_1:
4762 00005F8E 8A0B                     <1>       mov    cl, [ebx]
4763 00005F90 08C9                     <1>       or     cl, cl ; 0 = free, >0 = process number
4764 00005F92 7415                     <1>       jz     short swpout_timer_page_3
4765 00005F94 8B4B0C                   <1>       mov    ecx, [ebx+12] ; response (signal return) address
4766 00005F97 6681E100F0               <1>       and    cx, PTE_A_CLEAR ; clear offset part (right 12 bits)
4767                                    <1>                            ; of the response byte address, to
4768                                    <1>                            ; get beginning of the page address)
4769 00005F9C 39C8                     <1>       cmp    eax, ecx
4770 00005F9E 7505                     <1>       jne    short swpout_timer_page_2 ; not same page
4771                                    <1>
4772                                    <1>       ; !same page!
4773                                    <1>       ;
4774                                    <1>       ; NOTE: // 19/05/2016 // - TRDOS 386 feature only ! -
4775                                    <1>       ; This page will be used by the kernel to put timer event
4776                                    <1>       ; response (signal return) byte at the requested address;
4777                                    <1>       ; in order to prevent a possible wrong write (while
4778                                    <1>       ; this page is swapped out) on physical memory,
4779                                    <1>       ; we must protect this page against to be swapped out!
4780                                    <1>       ;
4781 00005FA0 5B                       <1>       pop    ebx ; ****
4782 00005FA1 59                       <1>       pop    ecx ; ***
4783 00005FA2 5A                       <1>       pop    edx ; **
4784 00005FA3 EBBF                     <1>       jmp    short swpout_1      ; do not swap out this page !
4785                                    <1>
4786                                    <1> swpout_timer_page_2:
4787                                    <1>       ; 07/06/2016
4788 00005FA5 FECA                     <1>       dec    dl
4789 00005FA7 7405                     <1>       jz     short swpout_timer_page_4
4790                                    <1> swpout_timer_page_3:
4791                                    <1>       ;cmp   ebx, timer_set + 240 ; last timer event (15*16)
4792                                    <1>       ;jnb   short swpout_timer_page_4
4793 00005FA9 83C310                   <1>       add    ebx, 16
4794 00005FAC EBE0                     <1>       jmp    short swpout_timer_page_1
4795                                    <1>
4796                                    <1> swpout_timer_page_4:
4797 00005FAE 5B                       <1>       pop    ebx ; ****
4798 00005FAF 59                       <1>       pop    ecx ; ***
4799                                    <1> swpout_2:
4800 00005FB0 89DA                     <1>       mov    edx, ebx            ; Page table entry address
4801 00005FB2 89C3                     <1>       mov    ebx, eax            ; Buffer (Page) Address
4802                                    <1>       ;
4803 00005FB4 E8A6010000               <1>       call   link_swap_block
4804 00005FB9 7304                     <1>       jnc    short swpout_3             ; It may not be needed here
4805                                    <1>                                   ; because [swpd_free] value
4806                                    <1>                                   ; was checked at the beginging.
4807 00005FBB 5A                       <1>       pop    edx ; **
4808 00005FBC 5B                       <1>       pop    ebx ; *
4809 00005FBD EB33                     <1>       jmp    short swpout_nfspc_err
4810                                    <1> swpout_3:
4811 00005FBF A900000080               <1>       test   eax, 80000000h ; test bit 31 (this may not be needed!)
4812 00005FC4 752C                     <1>       jnz    short swpout_nfspc_err  ; 10/06/2016 (bit 31 = 1 !)
4813                                    <1>       ;
4814 00005FC6 56                       <1>       push   esi ; **
4815 00005FC7 51                       <1>       push   ecx ; ***
4816 00005FC8 50                       <1>       push   eax ; sector address ; (31 bit !, bit 31 = 0)
4817 00005FC9 8B35[62050300]           <1>       mov    esi, [swp_drv]
4818 00005FCF B9080000000              <1>       mov    ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
4819                                    <1>              ; Note: Even if corresponding physical disk's sector
4820                                    <1>              ; size different than 512 bytes, logical disk sector
```

```
                                <1>             ; size is 512 bytes and disk writing procedure
                                <1>             ; will be performed for writing 4096 bytes
                                <1>             ; (2*2048, 8*512).
                                <1>       ; ESI = Logical disk description table address
                                <1>       ; EBX = Buffer (Page) address
                                <1>       ; EAX = Sector adress (offset address, logical sector number)
                                <1>       ; ECX = Sector count ; 8 sectors
                                <1>       ; edx = PTE address
00005FD4 E8E2010000             <1>       call   logical_disk_write
                                <1>       ; edx = PTE address
00005FD9 59                     <1>       pop    ecx ; sector address
00005FDA 730C                   <1>       jnc    short swpout_write_ok
                                <1>       ;
                                <1>       ;; call      unlink_swap_block ; this block must be left as 'in use'
                                <1> swpout_dw_err:
00005FDC B82C000000             <1>       mov    eax, SWP_DISK_WRITE_ERR ; drive not ready or write error
00005FE1 A3[C8030300]           <1>       mov    [u.error], eax
00005FE6 EB06                   <1>       jmp    short swpout_retn
                                <1>       ;
                                <1> swpout_write_ok:
                                <1>       ; EBX = Buffer (page) address
                                <1>       ; EDX = Page Table Entry address
                                <1>       ; ECX = Swap disk sector (file block) address (31 bit)
00005FE8 D1E1                   <1>       shl    ecx, 1  ; 31 bit sector address from bit 1 to bit 31
00005FEA 890A                   <1>       mov    [edx], ecx
                                <1>             ; bit 0 = 0 (swapped page)
00005FEC 89D8                   <1>       mov    eax, ebx
                                <1> swpout_retn:
00005FEE 59                     <1>       pop    ecx ; ***
00005FEF 5E                     <1>       pop    esi ; **
00005FF0 5B                     <1>       pop    ebx ; *
00005FF1 C3                     <1>       retn
                                <1>
                                <1> ;swpout_dnp_err:
                                <1> ;     mov    eax, SWP_DISK_NOT_PRESENT_ERR ; disk not present
                                <1> ;     jmp    short swpout_err_retn
                                <1> swpout_nfspc_err:
00005FF2 B82B000000             <1>       mov    eax, SWP_NO_FREE_SPACE_ERR ; no free space
                                <1> swpout_err_retn:
00005FF7 A3[C8030300]           <1>       mov    [u.error], eax
                                <1>       ;stc
00005FFC C3                     <1>       retn
                                <1> swpout_npts_err:
00005FFD B82D000000             <1>       mov    eax, SWP_NO_PAGE_TO_SWAP_ERR
00006002 5B                     <1>       pop    ebx
00006003 EBF2                   <1>       jmp    short swpout_err_retn
                                <1> swpout_im_err:
00006005 B804000000             <1>       mov    eax, ERR_MINOR_IM ; insufficient (out of) memory
0000600A EBEB                   <1>       jmp    short swpout_err_retn
                                <1>
                                <1> swap_queue_shift:
                                <1>       ; 26/03/2017
                                <1>       ; 10/06/2016
                                <1>       ; 09/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 23/10/2014 - 20/07/2015 (Retro UNIX 386 v1)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EBX = Virtual (linear) address (bit 12 to 31)
                                <1>       ;           and process number combination (bit 0 to 11)
                                <1>       ;     EBX = 0 -> shift/drop from the head (offset 0)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     If EBX input > 0
                                <1>       ;        the queue will be shifted 4 bytes (dword),
                                <1>       ;        from the tail to the head, up to entry offset
                                <1>       ;        which points to EBX input value or nothing
                                <1>       ;        to do if EBX value is not found on the queue.
                                <1>       ;        (The entry -with EBX value- will be removed
                                <1>       ;        from the queue if it is found.)
                                <1>       ;
                                <1>       ;        EAX = 0
                                <1>       ;
                                <1>       ;     If EBX input = 0
                                <1>       ;        the queue will be shifted 4 bytes (dword),
                                <1>       ;        from the tail to the head, if the PTE address
                                <1>       ;        which is pointed in head of the queue is marked
                                <1>       ;        as "accessed" or it is marked as "non present".
                                <1>       ;        (If "accessed" flag of the PTE -which is pointed
                                <1>       ;        in the head- is set -to 1-, it will be reset
                                <1>       ;        -to 0- and then, the queue will be rotated
                                <1>       ;        -without dropping pointer of the PTE from
                                <1>       ;        the queue- for 4 bytes on head to tail direction.
                                <1>       ;        Pointer in the head will be moved into the tail,
                                <1>       ;        other PTEs will be shifted on head direction.)
                                <1>       ;
                                <1>       ;        Swap queue will be shifted up to the first
                                <1>       ;        'present' or 'non accessed' page will be found
                                <1>       ;        (as pointed) on the queue head (then it will be
                                <1>         ;          removed/dropped from the queue).
                                <1>       ;
                                <1>       ;        EAX (> 0) = PTE value of the page which is
                                <1>       ;             (it's pointer -virtual address-) dropped
                                <1>       ;             (removed) from swap queue.
                                <1>       ;        EBX = PTE address of the page (if EAX > 0)
                                <1>       ;              which is (it's pointer -virtual address-)
                                <1>       ;             dropped (removed) from swap queue.
                                <1>       ;
                                <1>       ;        EAX = 0 -> empty swap queue !
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EBX
                                <1>       ;
0000600C 0FB705[60050300]       <1>       movzx   eax, word [swpq_count]  ; Max. 1024
00006013 6621C0                 <1>       and    ax, ax
00006016 7431                   <1>       jz     short swpqs_retn
00006018 57                     <1>       push   edi
```

```
4926 00006019 56                  <1>       push   esi
4927 0000601A 51                  <1>       push   ecx
4928 0000601B BE00E00800          <1>       mov    esi, swap_queue
4929 00006020 89C1                <1>       mov    ecx, eax
4930 00006022 09DB                <1>       or     ebx, ebx
4931 00006024 7424                <1>       jz     short swpqs_7
4932                              <1> swpqs_1:
4933 00006026 AD                  <1>       lodsd
4934 00006027 39D8                <1>       cmp    eax, ebx
4935 00006029 7406                <1>       je     short swpqs_2
4936 0000602B E2F9                <1>       loop   swpqs_1
4937                              <1>       ; 10/06/2016
4938 0000602D 29C0                <1>       sub    eax, eax
4939 0000602F EB15                <1>       jmp    short swpqs_6
4940                              <1> swpqs_2:
4941 00006031 89F7                <1>       mov    edi, esi
4942 00006033 83EF04              <1>       sub    edi, 4
4943                              <1> swpqs_3:
4944 00006036 66FF0D[60050300]    <1>       dec    word [swpq_count]
4945 0000603D 7403                <1>       jz     short swpqs_5
4946                              <1> swpqs_4:
4947 0000603F 49                  <1>       dec    ecx
4948 00006040 F3A5                <1>       rep    movsd  ; shift up (to the head)
4949                              <1> swpqs_5:
4950 00006042 31C0                <1>       xor    eax, eax
4951 00006044 8907                <1>       mov    [edi], eax
4952                              <1> swpqs_6:
4953 00006046 59                  <1>       pop    ecx
4954 00006047 5E                  <1>       pop    esi
4955 00006048 5F                  <1>       pop    edi
4956                              <1> swpqs_retn:
4957 00006049 C3                  <1>       retn
4958                              <1> swpqs_7:
4959 0000604A 89F7                <1>       mov    edi, esi ; head
4960 0000604C AD                  <1>       lodsd
4961                              <1>       ; 20/07/2015
4962 0000604D 89C3                <1>       mov    ebx, eax
4963 0000604F 81E300F0FFFF        <1>       and    ebx, ~PAGE_OFF ; ~0FFFh
4964                              <1>                    ; ebx = virtual address (at page boundary)
4965 00006055 25FF0F0000          <1>       and    eax, PAGE_OFF ; 0FFFh
4966                              <1>                    ; ax = process number (1 to 4095)
4967 0000605A 3A05[B3030300]      <1>       cmp    al, [u.uno]
4968                              <1>                    ; Max. 16 (nproc) processes for Retro UNIX 386 v1
4969 00006060 7507                <1>       jne    short swpqs_8
4970 00006062 A1[B8030300]        <1>       mov    eax, [u.pgdir]
4971 00006067 EB28                <1>       jmp    short swpqs_9
4972                              <1> swpqs_8:
4973                              <1>       ; 09/06/2016
4974 00006069 80B8[AF000300]00    <1>       cmp    byte [eax+p.stat-1], 0
4975 00006070 76C4                <1>       jna    short swpqs_3     ; free (or terminated) process
4976 00006072 80B8[AF000300]02    <1>       cmp    byte [eax+p.stat-1], 2 ; waiting
4977 00006079 77BB                <1>       ja     short swpqs_3         ; zombie (3) or undefined ?
4978                              <1>
4979                              <1>       ;shl   ax, 2
4980 0000607B C0E002              <1>       shl    al, 2
4981 0000607E 8B80[BC000300]      <1>       mov    eax, [eax+p.upage-4]
4982 00006084 09C0                <1>       or     eax, eax
4983 00006086 74AE                <1>       jz     short swpqs_3 ; invalid upage
4984 00006088 83C05C              <1>       add    eax, u.pgdir - user
4985                              <1>                     ; u.pgdir value for the process
4986                              <1>                     ; is in [eax]
4987 0000608B 8B00                <1>       mov    eax, [eax]
4988 0000608D 21C0                <1>       and    eax, eax
4989 0000608F 74A5                <1>       jz     short swpqs_3 ; invalid page directory
4990                              <1> swpqs_9:
4991 00006091 52                  <1>       push   edx
4992                              <1>       ; eax = page directory
4993                              <1>       ; ebx = virtual address
4994 00006092 E82BFBFFFF          <1>       call   get_pte
4995 00006097 89D3                <1>       mov    ebx, edx     ; PTE address
4996 00006099 5A                  <1>       pop    edx
4997                              <1>       ; 10/06/2016
4998 0000609A 723A                <1>       jc     short swpqs_13 ; empty PDE
4999                              <1>       ; EAX = PTE value
5000 0000609C A801                <1>       test   al, PTE_A_PRESENT ; bit 0 = 1
5001 0000609E 7436                <1>       jz     short swpqs_13  ; Drop non-present page
5002                              <1>                              ; from the queue (head)
5003 000060A0 A802                <1>       test   al, PTE_A_WRITE     ; bit 1 = 0 (read only)
5004 000060A2 7432                <1>       jz     short swpqs_13  ; Drop read only page
5005                              <1>                              ; from the queue (head)
5006                              <1>       ;test  al, PTE_A_ACCESS ; bit 5 = 1 (Accessed)
5007                              <1>       ;jnz   short swpqs_11  ; present
5008                              <1>                              ; accessed page
5009 000060A4 0FBAF005            <1>         btr     eax, PTE_A_ACCESS_BIT ; reset 'accessed' bit
5010 000060A8 7210                <1>       jc     short swpqs_11  ; accessed page
5011                              <1>
5012 000060AA 49                  <1>       dec    ecx
5013 000060AB 66890D[60050300]    <1>       mov    [swpq_count], cx
5014 000060B2 7402                <1>         jz      short swpqs_10
5015                              <1>             ; esi = head + 4
5016                              <1>             ; edi = head
5017 000060B4 F3A5                <1>       rep    movsd   ; n = 1 to k-1, [n - 1] = [n]
5018                              <1> swpqs_10:
5019 000060B6 890F                <1>       mov    [edi], ecx ; 0
5020 000060B8 EB8C                <1>       jmp    short swpqs_6 ; 26/03/2017
5021                              <1>
5022                              <1> swpqs_11:
5023 000060BA 8903                <1>       mov    [ebx], eax     ; save changed attribute
5024                              <1>       ; Rotation (head -> tail)
5025 000060BC 49                  <1>       dec    ecx     ; entry count -> last entry number
5026 000060BD 74F7                <1>       jz     short swpqs_10
5027                              <1>             ; esi = head + 4
5028                              <1>             ; edi = head
5029 000060BF 8B07                <1>       mov    eax, [edi] ; 20/07/2015
5030 000060C1 F3A5                <1>       rep    movsd   ; n = 1 to k-1, [n - 1] = [n]
```

```
5031 000060C3 8907                <1>       mov    [edi], eax ; head -> tail ; [k] = [1]
5032                              <1>
5033 000060C5 668B0D[60050300]    <1>       mov    cx, [swpq_count]
5034                              <1>
5035                              <1> swpqs_12:
5036 000060CC BE00E00800          <1>       mov    esi, swap_queue ; head
5037 000060D1 E974FFFFFF          <1>         jmp     swpqs_7
5038                              <1>
5039                              <1> swpqs_13:
5040 000060D6 49                  <1>       dec    ecx
5041 000060D7 66890D[60050300]    <1>       mov    [swpq_count], cx
5042 000060DE 0F845EFFFFFF        <1>         jz      swpqs_5
5043 000060E4 EBE6                <1>       jmp    short swpqs_12
5044                              <1>
5045                              <1> add_to_swap_queue:
5046                              <1> ; temporary - 16/09/2015
5047 000060E6 C3                  <1> retn
5048                              <1>       ; 20/02/2017
5049                              <1>       ; 20/07/2015
5050                              <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
5051                              <1>       ;
5052                              <1>       ; Adds new page to swap queue
5053                              <1>       ; (page directories and page tables must not be added
5054                              <1>       ; to swap queue)
5055                              <1>       ;
5056                              <1>       ; INPUT ->
5057                              <1>       ;      EBX = Linear (Virtual) addr for current process
5058                              <1>       ;      [u.uno]
5059                              <1>       ;      20/02/2017
5060                              <1>       ;      (Linear address = CORE + user's virtual address)
5061                              <1>       ;
5062                              <1>       ; OUTPUT ->
5063                              <1>       ;      EAX = [swpq_count]
5064                              <1>       ;            (after the PTE has been added)
5065                              <1>       ;      EAX = 0 -> Swap queue is full, (1024 entries)
5066                              <1>       ;            the PTE could not be added.
5067                              <1>       ;
5068                              <1>       ; Modified Registers -> EAX
5069                              <1>       ;
5070 000060E7 53                  <1>       push   ebx
5071 000060E8 6681E300F0          <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
5072 000060ED 8A1D[B3030300]      <1>       mov    bl, [u.uno] ; current process number
5073 000060F3 E814FFFFFF          <1>       call   swap_queue_shift ; drop from the queue if
5074                              <1>                              ; it is already on the queue
5075                              <1>             ; then add it to the tail of the queue
5076 000060F8 0FB705[60050300]    <1>       movzx  eax, word [swpq_count]
5077 000060FF 663D0004            <1>       cmp    ax, 1024
5078 00006103 7205                <1>       jb     short atsq_1
5079 00006105 6629C0              <1>       sub    ax, ax
5080 00006108 5B                  <1>       pop    ebx
5081 00006109 C3                  <1>       retn
5082                              <1> atsq_1:
5083 0000610A 56                  <1>       push   esi
5084 0000610B BE00E00800          <1>       mov    esi, swap_queue
5085 00006110 6621C0              <1>       and    ax, ax
5086 00006113 740A                <1>       jz     short atsq_2
5087 00006115 66C1E002            <1>       shl    ax, 2  ; convert to offset
5088 00006119 01C6                <1>       add    esi, eax
5089 0000611B 66C1E802            <1>       shr    ax, 2
5090                              <1> atsq_2:
5091 0000611F 6640                <1>       inc    ax
5092 00006121 891E                <1>       mov    [esi], ebx ; Virtual address + [u.uno] combination
5093 00006123 66A3[60050300]      <1>       mov    [swpq_count], ax
5094 00006129 5E                  <1>       pop    esi
5095 0000612A 5B                  <1>       pop    ebx
5096 0000612B C3                  <1>       retn
5097                              <1>
5098                              <1> unlink_swap_block:
5099                              <1>       ; 15/09/2015
5100                              <1>       ; 30/04/2015
5101                              <1>       ; 18/04/2015
5102                              <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
5103                              <1>       ;
5104                              <1>       ; INPUT ->
5105                              <1>       ;      EAX = swap disk/file offset address
5106                              <1>       ;            (bit 1 to bit 31)
5107                              <1>       ; OUTPUT ->
5108                              <1>       ;      [swpd_free] is increased
5109                              <1>       ;      (corresponding SWAP DISK ALLOC. TABLE bit is SET)
5110                              <1>       ;
5111                              <1>       ; Modified Registers -> EAX
5112                              <1>       ;
5113 0000612C 53                  <1>       push   ebx
5114 0000612D 52                  <1>       push   edx
5115                              <1>       ;
5116 0000612E C1E804              <1>       shr    eax, SECTOR_SHIFT+1  ;3+1 ; shift sector address to
5117                              <1>                                   ; 3 bits right
5118                              <1>                                   ; to get swap block/page number
5119 00006131 89C2                <1>       mov    edx, eax
5120                              <1>       ; 15/09/2015
5121 00006133 C1EA03              <1>       shr    edx, 3            ; to get offset to S.A.T.
5122                              <1>                                   ; (1 allocation bit = 1 page)
5123                              <1>                                   ; (1 allocation bytes = 8 pages)
5124 00006136 80E2FC              <1>       and    dl, 0FCh          ; clear lower 2 bits
5125                              <1>                                   ; (to get 32 bit position)
5126                              <1>       ;
5127 00006139 BB00000D00          <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table address
5128 0000613E 01D3                <1>       add    ebx, edx
5129 00006140 83E01F              <1>       and    eax, 1Fh          ; lower 5 bits only
5130                              <1>                                   ; (allocation bit position)
5131 00006143 3B05[6E050300]      <1>       cmp    eax, [swpd_next]     ; is the new free block addr. lower
5132                              <1>                                   ; than the address in 'swpd_next' ?
5133                              <1>                                   ; (next/first free block value)
5134 00006149 7305                <1>       jnb    short uswpbl_1           ; no
5135 0000614B A3[6E050300]        <1>       mov    [swpd_next], eax     ; yes
```

```
                                <1> uswpbl_1:
00006150 0FAB03                 <1>       bts    [ebx], eax        ; unlink/release/deallocate block
                                <1>                                ; set relevant bit to 1.
                                <1>                                ; set CF to the previous bit value
00006153 F5                     <1>       cmc                      ; complement carry flag
00006154 7206                   <1>       jc     short uswpbl_2          ; do not increase swfd_free count
                                <1>                                ; if the block is already deallocated
                                <1>                                ; before.
00006156 FF05[6A050300]         <1>         inc     dword [swpd_free]
                                <1> uswpbl_2:
0000615C 5A                     <1>       pop    edx
0000615D 5B                     <1>       pop    ebx
0000615E C3                     <1>       retn
                                <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>       ;
0000615F 53                     <1>       push   ebx
00006160 51                     <1>       push   ecx
                                <1>       ;
00006161 BB00000D00             <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table offset
00006166 89D9                   <1>       mov    ecx, ebx
00006168 031D[6E050300]         <1>       add    ebx, [swpd_next] ; Free block searching starts from here
                                <1>                             ; next_free_swap_block >> 5
0000616E 030D[72050300]         <1>       add    ecx, [swpd_last] ; Free block searching ends here
                                <1>                             ; (total_swap_blocks - 1) >> 5
                                <1> lswbl_scan:
00006174 39CB                   <1>       cmp    ebx, ecx
00006176 770A                   <1>       ja     short lswbl_notfound
                                <1>       ;
00006178 0FBC03                 <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
0000617B 751C                   <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!)
0000617D 83C304                 <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.
00006180 EBF2                   <1>       jmp          short lswbl_scan
                                <1>       ;
                                <1> lswbl_notfound:
00006182 81E900000D00           <1>       sub    ecx, swap_alloc_table
00006188 890D[6E050300]         <1>       mov    [swpd_next], ecx ; next/first free page = last page
                                <1>                               ; (unlink_swap_block procedure will change it)
0000618E 31C0                   <1>       xor    eax, eax
00006190 A3[6A050300]           <1>       mov    [swpd_free], eax
00006195 F9                     <1>       stc
                                <1> lswbl_ok:
00006196 59                     <1>       pop    ecx
00006197 5B                     <1>       pop    ebx
00006198 C3                     <1>       retn
                                <1>       ;
                                <1> ;out_of_swpspc:
                                <1> ;     stc
                                <1> ;     retn
                                <1>
                                <1> lswbl_found:
00006199 89D9                   <1>       mov    ecx, ebx
0000619B 81E900000D00           <1>       sub    ecx, swap_alloc_table
000061A1 890D[6E050300]         <1>       mov    [swpd_next], ecx ; Set first free block searching start
                                <1>                               ; address/offset (to the next)
000061A7 FF0D[6A050300]         <1>         dec     dword [swpd_free] ; 1 block has been allocated (X = X-1)
                                <1>       ;
000061AD 0FB303                 <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.
000061B0 C1E105                 <1>       shl    ecx, 5        ; (block offset * 32) + block index
000061B3 01C8                   <1>       add    eax, ecx      ; = block number
000061B5 C1E003                 <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>       ;
000061B8 EBDC                   <1>       jmp    short lswbl_ok
                                <1>
                                <1> logical_disk_read:
                                <1>       ; 20/07/2015
                                <1>       ; 09/03/2015 (temporary code here)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      ESI = Logical disk description table address
                                <1>       ;      EBX = Memory page (buffer) address (physical!)
                                <1>       ;      EAX = Sector adress (offset address, logical sector number)
                                <1>       ;      ECX = Sector count
                                <1>       ;
                                <1>       ;
000061BA C3                     <1>       retn
                                <1>
                                <1> logical_disk_write:
                                <1>       ; 20/07/2015
                                <1>       ; 09/03/2015 (temporary code here)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      ESI = Logical disk description table address
                                <1>       ;      EBX = Memory page (buffer) address (physical!)
                                <1>       ;      EAX = Sector adress (offset address, logical sector number)
                                <1>       ;      ECX = Sector count
                                <1>       ;
000061BB C3                     <1>       retn
                                <1>
                                <1> get_physical_addr:
                                <1>       ; 26/03/2017
                                <1>       ; 20/02/2017
                                <1>       ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 18/10/2015
                                <1>       ; 29/07/2015
                                <1>       ; 20/07/2015
                                <1>       ; 04/06/2015
                                <1>       ; 20/05/2015
                                <1>       ; 28/04/2015
                                <1>       ; 18/04/2015
                                <1>       ; Get physical address
                                <1>       ;     (allocates a new page for user if it is not present)
                                <1>       ;
                                <1>       ; (This subroutine is needed for mapping user's virtual
                                <1>       ; (buffer) address to physical address (of the buffer).)
                                <1>       ; ('sys write', 'sys read' system calls...)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EBX = virtual address
                                <1>       ;     u.pgdir = page directory (physical) address
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     EAX = physical address
                                <1>       ;     EBX = linear address
                                <1>       ;     EDX = physical address of the page frame
                                <1>       ;           (with attribute bits)
                                <1>       ;     ECX = byte count within the page frame
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EBX, ECX, EDX
                                <1>       ;
000061BC 81C300004000           <1>       add    ebx, CORE ; 18/10/2015
                                <1> get_physical_addr_x: ; 27/05/2016
000061C2 A1[B8030300]           <1>       mov    eax, [u.pgdir]
000061C7 E8F6F9FFFF             <1>       call   get_pte
                                <1>              ; EDX = Page table entry address (if CF=0)
                                <1>                ;       Page directory entry address (if CF=1)
                                <1>              ;       (Bit 0 value is 0 if PT is not present)
                                <1>              ; EAX = Page table entry value (page address)
                                <1>              ;     CF = 1 -> PDE not present or invalid ?
000061CC 731C                   <1>       jnc    short gpa_1
                                <1>       ;
000061CE E8D4F8FFFF             <1>       call   allocate_page
000061D3 7248                   <1>       jc     short gpa_im_err  ; 'insufficient memory' error
                                <1> gpa_0:
000061D5 E847F9FFFF             <1>       call   clear_page
                                <1>       ; EAX = Physical (base) address of the allocated (new) page
000061DA 0C07                   <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
                                <1>                        ; lower 3 bits are used as U/S, R/W, P flags
                                <1>                        ; (user, writable, present page)
000061DC 8902                   <1>       mov    [edx], eax ; Let's put the new page directory entry here !
000061DE A1[B8030300]           <1>       mov    eax, [u.pgdir]
000061E3 E8DAF9FFFF             <1>       call   get_pte
000061E8 7233                   <1>       jc     short gpa_im_err ; 'insufficient memory' error
                                <1> gpa_1:
                                <1>       ; EAX = PTE value, EDX = PTE address
000061EA A801                   <1>       test   al, PTE_A_PRESENT
000061EC 751F                   <1>       jnz    short gpa_3 ; 26/03/2017
000061EE 09C0                   <1>       or     eax, eax
000061F0 7456                   <1>       jz     short gpa_7  ; Allocate a new page
                                <1>       ; 20/07/2015
000061F2 55                     <1>       push   ebp
000061F3 89DD                   <1>       mov    ebp, ebx ; virtual (linear) address
                                <1>       ; reload swapped page
000061F5 E878000000             <1>       call   reload_page ; 28/04/2015
000061FA 5D                     <1>       pop    ebp
000061FB 724A                   <1>       jc     short gpa_retn
                                <1> gpa_2:
                                <1>       ; 26/03/2017
                                <1>       ; 20/02/2017
                                <1>       ; If a page will contain a Signal Response Byte
                                <1>       ; it must not be swapped out, because
                                <1>       ; timer service or irq callback service
                                <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>       ;
000061FD 803D[C29A0100]00       <1>       cmp    byte [no_page_swap], 0
00006204 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>       ; imediately for other callers (*)
                                <1>       ; (otherwise, swap queue would not be long enough)
00006206 E84B000000             <1>       call   gpa_8 ; 26/03/2017
0000620B EB1D                   <1>       jmp    short gpa_5
                                <1> gpa_3:
                                <1>       ; 26/03/2017
0000620D 803D[C29A0100]00       <1>       cmp    byte [no_page_swap], 0
00006214 7618                   <1>       jna    short gpa_6 ; this page can be swapped out
00006216 E83B000000             <1>       call   gpa_8
0000621B EB11                   <1>       jmp    short gpa_6
                                <1>
                                <1> gpa_im_err:
0000621D B804000000             <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
                                <1>                                ; Major error = 0 (No protection fault)
00006222 C3                     <1>       retn
                                <1> gpa_4:
                                <1>       ; 20/07/2015
                                <1>       ; 20/05/2015
                                <1>       ; add this page to swap queue
00006223 50                     <1>       push   eax
                                <1>       ; EBX = Linear (CORE+virtual) address ; 20/02/2017
00006224 E8BDFEFFFF             <1>       call   add_to_swap_queue
00006229 58                     <1>       pop    eax
                                <1> gpa_5:
                                <1>             ; PTE address in EDX
                                <1>             ; virtual address in EBX
                                <1>       ; EAX = memory page address
0000622A 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
0000622C 8902                   <1>       mov    [edx], eax  ; Update PTE value
                                <1> gpa_6:
                                <1>       ; 18/10/2015
0000622E 89D9                   <1>       mov    ecx, ebx
00006230 81E1FF0F0000           <1>       and    ecx, PAGE_OFF
00006236 89C2                   <1>       mov    edx, eax
00006238 662500F0               <1>       and    ax, PTE_A_CLEAR
0000623C 01C8                   <1>       add    eax, ecx
0000623E F7D9                   <1>       neg    ecx ; 1 -> -1 (0FFFFFFFFh), 4095 (0FFFh) -> -4095
00006240 81C100100000           <1>       add    ecx, PAGE_SIZE
00006246 F8                     <1>       clc
                                <1> gpa_retn:
00006247 C3                     <1>       retn
                                <1> gpa_7:
00006248 E85AF8FFFF             <1>       call   allocate_page
0000624D 72CE                   <1>       jc     short gpa_im_err ; 'insufficient memory' error
0000624F E8CDF8FFFF             <1>       call   clear_page
00006254 EBA7                   <1>       jmp    short gpa_2
                                <1>
                                <1> gpa_8: ; 26/03/2017
00006256 C605[C29A0100]00       <1>       mov    byte [no_page_swap], 0
0000625D 53                     <1>       push   ebx
0000625E 50                     <1>       push   eax ; 26/03/2017
0000625F 6681E300F0             <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
00006264 8A1D[B3030300]         <1>       mov    bl, [u.uno] ; current process number
0000626A E89DFDFFFF             <1>       call   swap_queue_shift ; drop from the queue if
                                <1>                               ; it is already on the queue
0000626F 58                     <1>       pop    eax ; 26/03/2017
00006270 5B                     <1>       pop    ebx
00006271 C3                     <1>       retn
                                <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>       ;
00006272 D1E8                   <1>       shr    eax, 1   ; Convert PTE value to swap disk address
00006274 53                     <1>       push   ebx      ;
00006275 89C3                   <1>       mov    ebx, eax ; Swap disk (offset) address
00006277 E82BF8FFFF             <1>       call   allocate_page
0000627C 720C                   <1>       jc     short rlp_im_err
0000627E 93                     <1>       xchg   eax, ebx
                                <1>       ; EBX = Physical memory (page) address
                                <1>       ; EAX = Swap disk (offset) address
                                <1>       ; EBP = Virtual (linear) memory address
0000627F E862FCFFFF             <1>       call   swap_in
00006284 720B                   <1>       jc     short rlp_swp_err  ; (swap disk/file read error)
00006286 89D8                   <1>       mov    eax, ebx
                                <1> rlp_retn:
00006288 5B                     <1>       pop    ebx
00006289 C3                     <1>       retn
                                <1>
                                <1> rlp_im_err:
0000628A B804000000             <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
```

```
                                <1>                             ; Major error = 0 (No protection fault)
0000628F EBF7                   <1>       jmp    short rlp_retn
                                <1>
                                <1> rlp_swp_err:
00006291 B828000000             <1>       mov    eax, SWP_DISK_READ_ERR ; Swap disk read error !
00006296 EBF0                   <1>       jmp    short rlp_retn
                                <1>
                                <1>
                                <1> copy_page_dir:
                                <1>       ; 19/09/2015
                                <1>       ; temporary - 07/09/2015
                                <1>       ; 07/09/2015 (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>       ;
00006298 E80AF8FFFF             <1>       call   allocate_page
0000629D 723E                   <1>       jc     short cpd_err
                                <1>       ;
0000629F 55                     <1>       push   ebp ; 20/07/2015
000062A0 56                     <1>       push   esi
000062A1 57                     <1>       push   edi
000062A2 53                     <1>       push   ebx
000062A3 51                     <1>       push   ecx
000062A4 8B35[B8030300]         <1>       mov    esi, [u.pgdir]
000062AA 89C7                   <1>       mov    edi, eax
000062AC 50                     <1>       push   eax ; save child's page directory address
                                <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)
000062AD A5                     <1>       movsd
000062AE BD00004000             <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
000062B3 B9FF030000             <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
                                <1> cpd_0:
000062B8 AD                     <1>       lodsd
                                <1>       ;or    eax, eax
                                <1>         ;jnz     short cpd_1
000062B9 A801                   <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
000062BB 7508                   <1>       jnz    short cpd_1
                                <1>       ; (virtual address at the end of the page table)
000062BD 81C500004000           <1>       add    ebp, 1024*4096 ; page size * PTE count
000062C3 EB0F                   <1>       jmp    short cpd_2
                                <1> cpd_1:
000062C5 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
000062C9 89C3                   <1>       mov    ebx, eax
                                <1>       ; EBX = Parent's page table address
000062CB E81F000000             <1>       call   copy_page_table
000062D0 720C                   <1>       jc     short cpd_p_err
                                <1>       ; EAX = Child's page table address
000062D2 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> cpd_2:
000062D4 AB                     <1>       stosd
000062D5 E2E1                   <1>       loop   cpd_0
                                <1>       ;
000062D7 58                     <1>       pop    eax  ; restore child's page directory address
                                <1> cpd_3:
000062D8 59                     <1>       pop    ecx
000062D9 5B                     <1>       pop    ebx
000062DA 5F                     <1>       pop    edi
000062DB 5E                     <1>       pop    esi
000062DC 5D                     <1>       pop    ebp
                                <1> cpd_err:
000062DD C3                     <1>       retn
                                <1> cpd_p_err:
                                <1>       ; release the allocated pages missing (recover free space)
000062DE 58                     <1>       pop    eax  ; the new page directory address (physical)
000062DF 8B1D[B8030300]         <1>       mov    ebx, [u.pgdir] ; parent's page directory address
000062E5 E8F6F8FFFF             <1>       call   deallocate_page_dir
000062EA 29C0                   <1>       sub    eax, eax ; 0
000062EC F9                     <1>       stc
000062ED EBE9                   <1>       jmp    short cpd_3
                                <1>
                                <1> copy_page_table:
                                <1>       ; 19/09/2015
                                <1>       ; temporary - 07/09/2015
                                <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
                                <1>       ;      EBP = page table entry index (from 'copy_page_dir')
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
                                <1>       ;      EBP = (recent) page table index (for 'add_to_swap_queue')
                                <1>       ;      CF = 1 -> error
                                <1>       ;
                                <1>       ; Modified Registers -> EBP (except EAX)
                                <1>       ;
000062EF E8B3F7FFFF             <1>       call   allocate_page
000062F4 725A                   <1>       jc     short cpt_err
                                <1>       ;
000062F6 50                     <1>       push   eax ; *
                                <1>       ;push  ebx
000062F7 56                     <1>       push   esi
000062F8 57                     <1>       push   edi
```

```
5556 000062F9 52                <1>       push   edx
5557 000062FA 51                <1>       push   ecx
5558                            <1>       ;
5559 000062FB 89DE              <1>       mov    esi, ebx
5560 000062FD 89C7              <1>       mov    edi, eax
5561 000062FF 89C2              <1>       mov    edx, eax
5562 00006301 81C200100000      <1>       add    edx, PAGE_SIZE
5563                            <1> cpt_0:
5564 00006307 AD                <1>       lodsd
5565 00006308 A801              <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
5566 0000630A 750B              <1>       jnz    short cpt_1
5567 0000630C 21C0              <1>       and    eax, eax
5568 0000630E 7430              <1>       jz     short cpt_2
5569                            <1>       ; ebp = virtual (linear) address of the memory page
5570 00006310 E85DFFFFFF        <1>       call   reload_page ; 28/04/2015
5571 00006315 7234              <1>       jc     short cpt_p_err
5572                            <1> cpt_1:
5573 00006317 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
5574 0000631B 89C1              <1>       mov    ecx, eax
5575                            <1>       ; Allocate a new page for the child process
5576 0000631D E885F7FFFF        <1>       call   allocate_page
5577 00006322 7227              <1>       jc     short cpt_p_err
5578 00006324 57                <1>       push   edi
5579 00006325 56                <1>       push   esi
5580 00006326 89CE              <1>       mov    esi, ecx
5581 00006328 89C7              <1>       mov    edi, eax
5582 0000632A B900040000        <1>       mov    ecx, PAGE_SIZE/4
5583 0000632F F3A5              <1>       rep    movsd  ; copy page (4096 bytes)
5584 00006331 5E                <1>       pop    esi
5585 00006332 5F                <1>       pop    edi
5586                            <1>       ;
5587 00006333 53                <1>       push   ebx
5588 00006334 50                <1>       push   eax
5589 00006335 89EB              <1>       mov    ebx, ebp
5590                            <1>       ; ebx = virtual address of the memory page
5591 00006337 E8AAFDFFFF        <1>       call   add_to_swap_queue
5592 0000633C 58                <1>       pop    eax
5593 0000633D 5B                <1>       pop    ebx
5594                            <1>       ;
5595                            <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
5596 0000633E 0C07              <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
5597                            <1> cpt_2:
5598 00006340 AB                <1>       stosd  ; EDI points to child's PTE
5599                            <1>       ;
5600 00006341 81C500100000      <1>       add    ebp, 4096 ; 20/07/2015 (next page)
5601                            <1>       ;
5602 00006347 39D7              <1>       cmp    edi, edx
5603 00006349 72BC              <1>       jb     short cpt_0
5604                            <1> cpt_p_err:
5605 0000634B 59                <1>       pop    ecx
5606 0000634C 5A                <1>       pop    edx
5607 0000634D 5F                <1>       pop    edi
5608 0000634E 5E                <1>       pop    esi
5609                            <1>       ;pop   ebx
5610 0000634F 58                <1>       pop    eax ; *
5611                            <1> cpt_err:
5612 00006350 C3                <1>       retn
5613                            <1>
5614                            <1> allocate_memory_block:
5615                            <1>       ; 01/05/2017
5616                            <1>       ; 28/04/2017
5617                            <1>       ; 25/04/2017
5618                            <1>       ; 01/04/2016, 02/04/2016, 03/04/2016
5619                            <1>       ; 13/03/2016, 14/03/2016
5620                            <1>       ; 12/03/2016 (TRDOS 386 = TRDOS v2.0)
5621                            <1>       ; Allocating contiguous memory pages (in the kernel's memory space)
5622                            <1>       ;
5623                            <1>       ; INPUT ->
5624                            <1>       ;      EAX = Beginning address (physical)
5625                            <1>       ;      EAX = 0 -> Allocate memory block from the first proper aperture
5626                            <1>       ;      ECX = Number of bytes to be allocated
5627                            <1>       ;
5628                            <1>       ; OUTPUT ->
5629                            <1>       ;      1) cf = 0 -> successful
5630                            <1>       ;      EAX = Beginning (physical) address of the allocated memory block
5631                            <1>       ;      ECX = Number of allocated bytes (rounded up to page borders)
5632                            <1>       ;      2) cf = 1 -> unsuccessful
5633                            <1>       ;       2.1) If EAX > 0 ->
5634                            <1>       ;            (Number of requested pages is more than # of free pages
5635                            <1>       ;             but contiguous free pages -the aperture- is not enough!)
5636                            <1>       ;            EAX = Beginning address of available aperture
5637                            <1>       ;                 (one of all aperture with max. aperture size/length)
5638                            <1>       ;            ECX = Size of available aperture (memory block) in bytes
5639                            <1>       ;       2.2) If EAX = 0 -> Out of memory error
5640                            <1>       ;                  (number of free pages is less than requested number)
5641                            <1>       ;            ECX = Total number of free bytes (free pages * 4096)
5642                            <1>       ;                 (It is not number of contiguous free bytes)
5643                            <1>       ;
5644                            <1>       ; (Modified Registers -> EAX, ECX)
5645                            <1>       ;
5646                            <1>       ; PURPOSE: Loading a file at memory for copying or running etc.
5647                            <1>       ; If this procedure returns with cf is set, ECX contains maximum
5648                            <1>       ; available space and EAX contains the beginning address of it.
5649                            <1>       ; If EAX has zero, ECX contains total number of free bytes.
5650                            <1>       ; If requested block has been successfully allocated (by rounding up to
5651                            <1>       ; the last page border), it must be deallocated later by using
5652                            <1>       ; 'deallocate_memory_block' procedure.
5653                            <1>
5654 00006351 52                <1>       push   edx ; *
5655 00006352 BAFF0F0000        <1>       mov    edx, PAGE_SIZE - 1   ; 4095
5656 00006357 01D0              <1>       add    eax, edx
5657 00006359 01D1              <1>       add    ecx, edx
5658 0000635B C1E90C            <1>       shr    ecx, PAGE_SHIFT         ; 12
5659                            <1>
5660                            <1>       ; ECX = number of contiguous pages to be allocated
```

```
5661 0000635E 8B15[F8880100]      <1>       mov    edx, [free_pages]
5662                              <1>       ; 01/05/2017
5663                              <1>       ;or    ecx, ecx
5664                              <1>       ;jz    short amb3
5665                              <1>       ; If ECX=0, set cf to 1 and return with max. available mem block size
5666                              <1>
5667 00006364 39D1                <1>       cmp    ecx, edx
5668 00006366 7760                <1>       ja     short amb_3
5669                              <1>
5670 00006368 C1E80C              <1>       shr    eax, PAGE_SHIFT      ; 12
5671                              <1>
5672 0000636B 89C2                <1>       mov    edx, eax          ; page number
5673 0000636D C1EA03              <1>       shr    edx, 3            ; to get offset to M.A.T.
5674                              <1>                                ; (1 allocation bit = 1 page)
5675                              <1>                                ; (1 allocation bytes = 8 pages)
5676 00006370 80E2FC              <1>       and    dl, 0FCh          ; clear lower 2 bits
5677                              <1>                                ; (to get 32 bit position)
5678 00006373 53                  <1>       push   ebx ; **
5679                              <1> amb_0:
5680 00006374 890D[AC940100]      <1>       mov    [mem_ipg_count], ecx ; initial (reset) value of page count
5681 0000637A 890D[B0940100]      <1>       mov    [mem_pg_count], ecx
5682 00006380 31C9                <1>       xor    ecx, ecx ; 0
5683 00006382 890D[B4940100]      <1>       mov    [mem_aperture], ecx ; 0
5684 00006388 890D[B8940100]      <1>       mov    [mem_max_aperture], ecx ; 0
5685                              <1>
5686 0000638E BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address.
5687 00006393 3B15[FC880100]      <1>       cmp    edx, [next_page]     ; Is the beginning page address lower
5688                              <1>                                ; than the address in 'next_page' ?
5689                              <1>                                ; (the first/next free page of user space)
5690 00006399 7208                <1>       jb     short amb_1
5691 0000639B 3B15[00890100]      <1>       cmp    edx, [last_page]     ; is the beginning page address higher
5692                              <1>                                ; than the address in 'last_page' ?
5693                              <1>                                ; (end of the memory)
5694 000063A1 7606                <1>       jna    short amb_2       ; no
5695                              <1> amb_1:
5696 000063A3 8B15[FC880100]      <1>       mov    edx, [next_page]     ; M.A.T. offset (1 M.A.T. byte = 8 pages)
5697                              <1> amb_2:
5698 000063A9 01D3                <1>       add    ebx, edx
5699                              <1>
5700                              <1>       ; 28/04/2017
5701                              <1>       ;xor   ecx, ecx
5702 000063AB 0FBC0B              <1>       bsf    ecx, [ebx]        ; 0 to 31
5703 000063AE 89D0                <1>       mov    eax, edx
5704 000063B0 C1E003              <1>       shl    eax, 3            ; *8
5705 000063B3 01C8                <1>       add    eax, ecx          ; beginning page number
5706                              <1>
5707 000063B5 A3[BC940100]        <1>       mov    [mem_pg_pos], eax    ; beginning page no (for curr. mem. aperture)
5708 000063BA A3[C0940100]        <1>       mov    [mem_max_pg_pos], eax ; beginning page no for max. mem. aperture
5709                              <1>
5710 000063BF 83E01F              <1>       and    eax, 1Fh          ; lower 5 bits only (0 to 31)
5711                              <1>                                ; (allocation bit position)
5712 000063C2 750E                <1>       jnz    short amb_4       ; 0
5713 000063C4 B120                <1>       mov    cl, 32
5714 000063C6 EB4B                <1>       jmp    short amb_10
5715                              <1>
5716                              <1> amb_3:      ; out_of_memory
5717 000063C8 31C0                <1>       xor    eax, eax ; 0
5718 000063CA 89D1                <1>       mov    ecx, edx ; free pages
5719 000063CC C1E10C              <1>       shl    ecx, PAGE_SHIFT
5720 000063CF 5A                  <1>       pop    edx ; *
5721 000063D0 F9                  <1>       stc
5722 000063D1 C3                  <1>       retn
5723                              <1> amb_4:
5724 000063D2 8B13                <1>       mov    edx, [ebx]
5725 000063D4 88C1                <1>       mov    cl, al ; 1 to 31
5726 000063D6 D3EA                <1>       shr    edx, cl
5727 000063D8 89D0                <1>       mov    eax, edx
5728                              <1> amb_5:
5729 000063DA D1E8                <1>       shr    eax, 1 ; (***)
5730 000063DC 7317                <1>       jnc    short amb_7
5731 000063DE FF05[B4940100]      <1>       inc    dword [mem_aperture]
5732 000063E4 FF0D[B0940100]      <1>       dec    dword [mem_pg_count]
5733 000063EA 7470                <1>       jz     short amb_15
5734                              <1> amb_6:
5735                              <1>       ; 28/04/2017
5736 000063EC FEC1                <1>       inc    cl
5737 000063EE 80F920              <1>       cmp    cl, 32
5738 000063F1 730D                <1>       jnb    short amb_9
5739 000063F3 EBE5                <1>       jmp    short amb_5
5740                              <1> amb_7:
5741 000063F5 50                  <1>       push   eax ; (***) allocation bits (in shifted status)
5742 000063F6 E81B010000          <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
5743 000063FB 58                  <1>       pop    eax ; (***)
5744 000063FC EBEE                <1>       jmp    short amb_6
5745                              <1> amb_8:
5746                              <1>       ; 28/04/2017
5747 000063FE B120                <1>       mov    cl, 32
5748                              <1> amb_9:
5749 00006400 89DA                <1>       mov    edx, ebx
5750 00006402 81EA00001000        <1>       sub    edx, MEM_ALLOC_TBL
5751 00006408 3B15[00890100]      <1>       cmp    edx, [last_page]
5752 0000640E 7336                <1>       jnb    short amb_14 ; contiguous pages not enough
5753 00006410 83C304              <1>       add    ebx, 4
5754                              <1> amb_10:
5755 00006413 8B03                <1>       mov    eax, [ebx]
5756 00006415 21C0                <1>       and    eax, eax
5757 00006417 7408                <1>         jz     short amb_11 ; there is not a free page bit in this alloc dword
5758 00006419 40                  <1>       inc    eax ; 0FFFFFFFFh -> 0
5759 0000641A 740C                <1>       jz     short amb_12 ; all of bits are set (32 free pages)
5760 0000641C 48                  <1>       dec    eax
5761 0000641D 28C9                <1>       sub    cl, cl ; 0
5762 0000641F EBB9                <1>       jmp    short amb_5
5763                              <1> amb_11:
5764 00006421 E8F0000000          <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
5765 00006426 EBD8                <1>       jmp    short amb_9
```

```
                                <1> amb_12:
00006428 390D[B0940100]         <1>         cmp    [mem_pg_count], ecx ; 32
0000642E 7306                   <1>         jnb    short amb_13
00006430 8B0D[B0940100]         <1>         mov    ecx, [mem_pg_count]
                                <1> amb_13:
00006436 010D[B4940100]         <1>         add    [mem_aperture], ecx
0000643C 290D[B0940100]         <1>         sub    [mem_pg_count], ecx
00006442 7618                   <1>         jna    short amb_15
00006444 EBBA                   <1>         jmp    short amb_9 ; 01/05/2017
                                <1> amb_14:
00006446 E8CB000000             <1>         call   amb_26 ; 28/04/2017
0000644B A1[C0940100]           <1>         mov    eax, [mem_max_pg_pos] ; begin address of max. mem aperture
00006450 8B0D[B8940100]         <1>         mov    ecx, [mem_max_aperture] ; max. (largest) memory aperture
00006456 F9                     <1>         stc
00006457 E9AF000000             <1>           jmp     amb_25
                                <1>
                                <1> amb_15: ; OK !
0000645C A1[BC940100]           <1>         mov    eax, [mem_pg_pos]    ; Beginning address as page number
00006461 8B0D[B4940100]         <1>         mov    ecx, [mem_aperture]  ; Free contiguous page count (>=1)
                                <1> amb_16:
                                <1>         ; allocate contiguous memory pages (via memory allocation table bits)
00006467 89C2                   <1>         mov    edx, eax
                                <1>         ; 25/04/2017
00006469 C1EA03                 <1>         shr    edx, 3        ; 8 pages in one allocation byte
0000646C 80E2FC                 <1>         and    dl, 0FCh      ; clear lower 2 bits
                                <1>                              ; (for dword/32bit positioning)
                                <1>
0000646F BB00001000             <1>         mov    ebx, MEM_ALLOC_TBL
00006474 01D3                   <1>         add    ebx, edx
00006476 83E01F                 <1>         and    eax, 1Fh ; 31
                                <1>         ; 03/04/2016
00006479 BA20000000             <1>         mov    edx, 32
0000647E 28C2                   <1>         sub    dl, al
00006480 39CA                   <1>         cmp    edx, ecx      ; ecx >= 1
00006482 7602                   <1>         jna    short amb_17
00006484 89CA                   <1>         mov    edx, ecx
                                <1> amb_17:
00006486 29D1                   <1>         sub    ecx, edx
00006488 51                     <1>         push   ecx ; ***
00006489 89D1                   <1>         mov    ecx, edx
                                <1> amb_18:
0000648B 0FB303                 <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.
0000648E FF0D[F8880100]         <1>         dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
00006494 49                     <1>         dec    ecx
00006495 7404                   <1>         jz     short amb_19
00006497 FEC0                   <1>         inc    al
00006499 EBF0                   <1>         jmp    short amb_18
                                <1> amb_19:
0000649B 59                     <1>         pop    ecx ; ***
0000649C 21C9                   <1>         and    ecx, ecx ; 0 ?
0000649E 741E                   <1>         jz     short amb_22
                                <1>         ; 01/04/2016
000064A0 B020                   <1>         mov    al, 32
                                <1> amb_20:
000064A2 83C304                 <1>         add    ebx, 4
000064A5 39C1                   <1>         cmp    ecx, eax ; 32
000064A7 7305                   <1>         jnb    short amb_21
                                <1>         ; ECX < 32
000064A9 28C0                   <1>         sub    al, al ; 0
000064AB 50                     <1>         push   eax ; 0 ***
000064AC EBDD                   <1>         jmp    short amb_18
                                <1> amb_21:
000064AE 2905[F8880100]         <1>         sub    [free_pages], eax   ; [free_pages] = [free_pages] - 32
000064B4 C70300000000           <1>         mov    dword [ebx], 0          ; reset 32 bits
000064BA 29C1                   <1>         sub    ecx, eax ; 32
000064BC 75E4                   <1>         jnz    short amb_20
                                <1> amb_22:
000064BE A1[BC940100]           <1>         mov    eax, [mem_pg_pos]   ; Beginning address as page number
000064C3 8B0D[B4940100]         <1>         mov    ecx, [mem_aperture] ; Free contiguous page count
                                <1>         ; [next_page] update
000064C9 89C2                   <1>         mov    edx, eax
                                <1>         ; 03/04/2016
000064CB C1EA03                 <1>         shr    edx, 3            ; to get offset to M.A.T.
                                <1>                                  ; (1 allocation bit = 1 page)
                                <1>                                  ; (1 allocation bytes = 8 pages)
000064CE 80E2FC                 <1>         and    dl, 0FCh          ; clear lower 2 bits
                                <1>                                  ; (to get 32 bit position)
000064D1 3B15[FC880100]         <1>         cmp    edx, [next_page] ; first free page pointer offset
000064D7 7732                   <1>         ja     short amb_25
000064D9 BB00001000             <1>         mov    ebx, MEM_ALLOC_TBL
000064DE 833C1300               <1>         cmp    dword [ebx+edx], 0
000064E2 7721                   <1>         ja     short amb_24
000064E4 89C2                   <1>         mov    edx, eax
000064E6 01CA                   <1>         add    edx, ecx
000064E8 C1EA03                 <1>         shr    edx, 3
000064EB 80E2FC                 <1>         and    dl, 0FCh
                                <1> amb_23:
000064EE 833C1300               <1>         cmp    dword [ebx+edx], 0
000064F2 7711                   <1>         ja     short amb_24
000064F4 83C204                 <1>         add    edx, 4
000064F7 3B15[00890100]         <1>         cmp    edx, [last_page]    ; last page pointer offset
000064FD 76EF                   <1>         jna    short amb_23
000064FF 8B15[04890100]         <1>         mov    edx, [first_page]   ; (for) beginning of user's space
                                <1> amb_24:
00006505 8915[FC880100]         <1>         mov    [next_page], edx
                                <1> amb_25:
0000650B 9C                     <1>         pushf
0000650C C1E00C                 <1>         shl    eax, PAGE_SHIFT          ; convert to phy. address in bytes
0000650F C1E10C                 <1>         shl    ecx, PAGE_SHIFT          ; convert to byte counts
00006512 9D                     <1>         popf
00006513 5B                     <1>         pop    ebx ; **
00006514 5A                     <1>         pop    edx ; *
00006515 C3                     <1>         retn
```

```
                                <1>
                                <1> amb_26:     ; set maximum free memory aperture (free memory block size)
00006516 89DA                   <1>       mov   edx, ebx ; current address
00006518 81EA00001000           <1>       sub   edx, MEM_ALLOC_TBL ; MAT beginning address
                                <1>       ; 02/04/2016
0000651E C1E203                 <1>       shl   edx, 3 ; MAT byte offset * 8 = page number base
00006521 01CA                   <1>       add   edx, ecx ; current page number (ecx = 0 to 32)
                                <1>       ;
00006523 A1[B4940100]           <1>       mov   eax, [mem_aperture]
00006528 21C0                   <1>       and   eax, eax
0000652A 7421                   <1>         jz      short amb_27
0000652C C705[B4940100]0000-    <1>         mov     dword [mem_aperture], 0
00006534 0000                   <1>
00006536 3B05[B8940100]         <1>       cmp   eax, [mem_max_aperture]
0000653C 760F                   <1>       jna   short amb_27
0000653E A3[B8940100]           <1>       mov   [mem_max_aperture], eax
                                <1>       ; 25/04/2017
00006543 A1[BC940100]           <1>       mov   eax, [mem_pg_pos]
                                <1>       ; EAX = Beginning page number of the max. aperture
00006548 A3[C0940100]           <1>       mov   [mem_max_pg_pos], eax
                                <1> amb_27:
0000654D 8915[BC940100]         <1>       mov   [mem_pg_pos], edx ; current page
                                <1>
00006553 A1[AC940100]           <1>       mov   eax, [mem_ipg_count] ; initial (reset) value of page count
00006558 A3[B0940100]           <1>       mov   [mem_pg_count], eax
                                <1>
0000655D C3                     <1>       retn
                                <1>
                                <1> deallocate_memory_block:
                                <1>       ; 03/04/2016
                                <1>       ; 14/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; Deallocating contiguous memory pages (in the kernel's memory space)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EAX = Beginning address (physical)
                                <1>       ;     ECX = Number of bytes to be deallocated
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     Memory Allocation Table bits will be updated
                                <1>       ;     [free_pages] will be changed (increased)
                                <1>       ;
                                <1>       ; (Modified Registers -> EAX, ECX)
                                <1>       ;
                                <1>       ; PURPOSE: Unloading/Freeing a file -or an allocated memory block-
                                <1>       ; at memory after copying, running, saving, reading, writing etc.
                                <1>       ;
                                <1>
0000655E 52                     <1>       push  edx ; *
0000655F 53                     <1>       push  ebx ; **
                                <1>
00006560 C1E80C                 <1>       shr   eax, PAGE_SHIFT         ; 12
00006563 C1E90C                 <1>       shr   ecx, PAGE_SHIFT         ; 12
                                <1>
                                <1>       ; EAX = Beginning page number
                                <1>       ; ECX = Number of contiguous pages to be deallocated
                                <1> damb_0:
                                <1>       ; deallocate contiguous memory pages (via memory allocation table bits)
00006566 89C2                   <1>       mov   edx, eax
00006568 C1EA03                 <1>       shr   edx, 3            ; to get offset to M.A.T.
                                <1>                               ; (1 allocation bit = 1 page)
                                <1>                               ; (1 allocation bytes = 8 pages)
0000656B 80E2FC                 <1>       and   dl, 0FCh          ; clear lower 2 bits
                                <1>                               ; (to get 32 bit position)
0000656E 3B15[FC880100]         <1>       cmp   edx, [next_page] ; next free page
00006574 7306                   <1>       jnb   short damb_1
00006576 8915[FC880100]         <1>       mov   [next_page], edx
                                <1> damb_1:
0000657C BB00001000             <1>       mov   ebx, MEM_ALLOC_TBL
00006581 01D3                   <1>       add   ebx, edx
00006583 83E01F                 <1>       and   eax, 1Fh ; 31
                                <1>
                                <1>       ; 03/04/2016
00006586 BA20000000             <1>       mov   edx, 32
0000658B 28C2                   <1>       sub   dl, al
0000658D 39CA                   <1>       cmp   edx, ecx
0000658F 7602                   <1>       jna   short damb_2
00006591 89CA                   <1>       mov   edx, ecx
                                <1> damb_2:
00006593 29D1                   <1>       sub   ecx, edx
00006595 51                     <1>       push  ecx ; ***
00006596 89D1                   <1>       mov   ecx, edx
                                <1> damb_3:
00006598 0FAB03                 <1>       bts   [ebx], eax        ; unlink/release/deallocate page
                                <1>                               ; set relevant bit to 1.
                                <1>                               ; set CF to the previous bit value
0000659B FF05[F8880100]         <1>       inc     dword [free_pages]   ; 1 page has been deallocated (X = X+1)
000065A1 49                     <1>       dec   ecx
000065A2 7404                   <1>       jz    short damb_4
000065A4 FEC0                   <1>       inc   al
000065A6 EBF0                   <1>       jmp   short damb_3
                                <1> damb_4:
000065A8 59                     <1>       pop   ecx ; ***
000065A9 21C9                   <1>       and   ecx, ecx ; 0 ?
000065AB 741E                   <1>       jz    short damb_7
                                <1>       ; 03/04/2016
000065AD B020                   <1>       mov   al, 32
                                <1> damb_5:
000065AF 83C304                 <1>       add   ebx, 4
000065B2 39C1                   <1>       cmp   ecx, eax ; 32
000065B4 7305                   <1>       jnb   short damb_6
                                <1>       ; ECX < 32
000065B6 28C0                   <1>       sub   al, al ; 0
000065B8 50                     <1>       push  eax ; 0 ***
000065B9 EBDD                   <1>       jmp   short damb_3
                                <1> damb_6:
```

```
5975 000065BB 0105[F8880100]      <1>       add    [free_pages], eax ; [free_pages] = [free_pages] + 32
5976 000065C1 C703FFFFFFFF        <1>       mov    dword [ebx], 0FFFFFFFFh ; set 32 bits
5977 000065C7 29C1                <1>       sub    ecx, eax ; 32
5978 000065C9 75E4                <1>       jnz    short damb_5
5979                              <1> damb_7:
5980 000065CB 5B                  <1>       pop    ebx ; **
5981 000065CC 5A                  <1>       pop    edx ; *
5982 000065CD C3                  <1>       retn
5983                              <1>
5984                              <1> direct_memory_access:
5985                              <1>       ; 22/07/2017
5986                              <1>       ; 12/05/2017
5987                              <1>       ; 16/07/2016
5988                              <1>       ; 12/07/2016 (TRDOS 386 = TRDOS v2.0)
5989                              <1>       ; This processure will be called to map
5990                              <1>       ; user's (ring 3) page tables to access phsical
5991                              <1>       ; (flat/linear) memory addresses, directly (without
5992                              <1>       ;  kernel's data transfer functions).
5993                              <1>       ;
5994                              <1>       ; Purpose: Video memory access and shared memory access.
5995                              <1>       ;
5996                              <1>       ; INPUT ->
5997                              <1>       ;     EAX = Beginning address (physical).
5998                              <1>       ;     EBX = User's buffer address ; 12/05/2017
5999                              <1>       ;     ECX = Number of contiguous pages to be mapped.
6000                              <1>       ; OUTPUT ->
6001                              <1>       ;     User's page directory and pages tables
6002                              <1>       ;     will be updated.
6003                              <1>       ;
6004                              <1>       ;     If an old page table entry has valid page address,
6005                              <1>       ;     that page will be deallocated just before PTE will
6006                              <1>       ;     be changed for direct (1 to 1) memory page access.
6007                              <1>       ;
6008                              <1>       ;     If old PTE value points to a swapped page,
6009                              <1>         ;       that page (block) will be unlinked on swap disk.
6010                              <1>       ;
6011                              <1>       ;     Newly allocated pages (except page tables) will not
6012                              <1>       ;     be applied to Memory Allocation Table.
6013                              <1>       ;     AVL bit 1 (PTE bit 10) of page table entry will be
6014                              <1>       ;     used to indicate shared (direct) memory page; then,
6015                              <1>       ;     this page will not be deallocated later during
6016                              <1>       ;     process termination. (Memory Allocation Table and
6017                              <1>       ;     free memory count will not be affected.
6018                              <1>       ;     (Except deallocating page table's itself.)
6019                              <1>       ;
6020                              <1>       ;      CF = 1 -> error (EAX = error code)
6021                              <1>       ;      CF = 0 -> success (EAX = beginning address)
6022                              <1>       ;
6023                              <1>       ;; (Modified Registers -> none)
6024                              <1>       ; Modified registers: ebp, edx, ecx, ebx, esi, edi
6025                              <1>       ;
6026                              <1>
6027                              <1>       ;push  ebp
6028                              <1>       ;push  ebx
6029                              <1>       ;push  ecx
6030                              <1>       ;push  edx
6031 000065CE 662500F0           <1>       and    ax, PTE_A_CLEAR ; clear page offset
6032 000065D2 50                  <1>       push   eax
6033                              <1>       ;and   ecx, ecx ; page count
6034                              <1>       ;jz    dmem_acc_7  ; 'insufficient memory' error
6035 000065D3 89C5                <1>       mov    ebp, eax
6036 000065D5 81C300004000        <1>       add    ebx, CORE ; 12/05/2017
6037                              <1> dmem_acc_0:
6038 000065DB 891D[AC9F0100]      <1>       mov    [base_addr], ebx ; 12/05/2017
6039 000065E1 A1[B8030300]        <1>       mov    eax, [u.pgdir] ; page dir address (physical)
6040 000065E6 E8D7F5FFFF          <1>       call   get_pte
6041                              <1>              ; EDX = Page table entry address (if CF=0)
6042                              <1>                ;       Page directory entry address (if CF=1)
6043                              <1>              ;       (Bit 0 value is 0 if PT is not present)
6044                              <1>              ; EAX = Page table entry value (page address)
6045                              <1>              ;     CF = 1 -> PDE not present or invalid ?
6046 000065EB 7324                <1>       jnc    short dmem_acc_1
6047                              <1>       ;
6048 000065ED E8B5F4FFFF          <1>       call   allocate_page
6049 000065F2 0F82AB000000        <1>         jc     dmem_acc_7  ; 'insufficient memory' error
6050                              <1>       ;
6051 000065F8 E824F5FFFF          <1>       call   clear_page
6052                              <1>       ; EAX = Physical (base) address of the allocated (new) page
6053 000065FD 0C07                <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
6054                              <1>                          ; lower 3 bits are used as U/S, R/W, P flags
6055                              <1>                          ; (user, writable, present page)
6056 000065FF 8902                <1>       mov    [edx], eax ; Let's put the new page directory entry here !
6057 00006601 A1[B8030300]        <1>       mov    eax, [u.pgdir]
6058 00006606 E8B7F5FFFF          <1>       call   get_pte
6059 0000660B 0F8292000000        <1>         jc     dmem_acc_7 ; 'insufficient memory' error
6060                              <1> dmem_acc_1:
6061                              <1>       ; EAX = PTE value, EDX = PTE address
6062 00006611 A801                <1>       test   al, PTE_A_PRESENT
6063 00006613 750D                <1>       jnz    short dmem_acc_2
6064 00006615 09C0                <1>       or     eax, eax
6065 00006617 7468                <1>       jz     short dmem_acc_6   ; Change PTE
6066 00006619 D1E8                <1>       shr    eax, 1       ; swap disk block (8 sectors) address
6067                              <1>       ; unlink swap disk block
6068 0000661B E80CFBFFFF          <1>       call   unlink_swap_block
6069 00006620 EB5F                <1>       jmp    short dmem_acc_6
6070                              <1>
6071                              <1> dmem_acc_2:
6072 00006622 A802                <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
6073                              <1>                              ; (must be 1)
6074 00006624 7550                <1>       jnz    short dmem_acc_4
6075                              <1>       ; Read only -duplicated- page (belongs to a parent or a child)
6076 00006626 66A90002            <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
6077                              <1>                                ; as child's page ?
6078 0000662A 7455                <1>       jz     short dmem_acc_5 ; Change PTE but don't deallocate the page!
6079                              <1>
```

```
6080                                    <1>       ;push  edi
6081                                    <1>       ;push  esi
6082                                    <1>
6083 0000662C 51                        <1>       push   ecx
6084                                    <1>       ;push  ebx
6085 0000662D 8B1D[BC030300]            <1>       mov    ebx, [u.ppgdir] ; parent's page dir address (physical)
6086                                    <1>
6087                                    <1>       ; check the parent's PTE value is read only & same page or not..
6088 00006633 89EF                      <1>       mov    edi, ebp
6089 00006635 C1EF16                    <1>       shr    edi, PAGE_D_SHIFT ; 22
6090                                    <1>       ; EDI = page directory entry index (0-1023)
6091 00006638 89EE                      <1>       mov    esi, ebp
6092 0000663A C1EE0C                    <1>       shr    esi, PAGE_SHIFT ; 12
6093 0000663D 81E6FF030000              <1>       and    esi, PTE_MASK
6094                                    <1>       ; ESI = page table entry index (0-1023)
6095                                    <1>
6096 00006643 66C1E702                  <1>       shl    di, 2 ; * 4
6097 00006647 01FB                      <1>       add    ebx, edi ; PDE offset (for the parent)
6098 00006649 8B0F                      <1>       mov    ecx, [edi]
6099 0000664B F6C101                    <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
6100 0000664E 7425                      <1>       jz     short dmem_acc_3    ; parent process does not use this page
6101 00006650 6681E100F0                <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
6102 00006655 66C1E602                  <1>       shl    si, 2 ; *4
6103 00006659 01CE                      <1>       add    esi, ecx ; PTE offset (for the parent)
6104 0000665B 8B1E                      <1>       mov    ebx, [esi]
6105 0000665D F6C301                    <1>       test   bl, PTE_A_PRESENT ; present or not ?
6106 00006660 7413                      <1>       jz     short dmem_acc_3    ; parent process does not use this page
6107 00006662 662500F0                  <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6108 00006666 6681E300F0                <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6109 0000666B 39D8                      <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
6110 0000666D 7506                      <1>       jne    short dmem_acc_3    ; not same page
6111                                    <1>                            ; deallocate the child's page
6112 0000666F 800E02                    <1>         or     byte [esi], PTE_A_WRITE ; convert to writable page (parent)
6113                                    <1>       ;pop   ebx
6114 00006672 59                        <1>       pop    ecx
6115 00006673 EB0C                      <1>       jmp    short dmem_acc_5
6116                                    <1> dmem_acc_3:
6117                                    <1>       ;pop   ebx
6118 00006675 59                        <1>       pop    ecx
6119                                    <1> dmem_acc_4:
6120 00006676 66A90004                  <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
6121 0000667A 7505                      <1>       jnz    short dmem_acc_5   ; AVL bit 1 = 1, do not deallocate this page!
6122                                    <1>       ;
6123                                    <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
6124 0000667C E804F6FFFF                <1>       call   deallocate_page
6125                                    <1> dmem_acc_5:
6126                                    <1>       ;pop   esi
6127                                    <1>       ;pop   edi
6128                                    <1> dmem_acc_6:
6129 00006681 89E8                      <1>       mov    eax, ebp ; physical page (offset=0) address
6130                                    <1>       ; EAX = memory page address
6131                                    <1>       ; EDX = PTE entry address (physical)
6132 00006683 660D0704                  <1>       or     ax, PTE_A_PRESENT+PTE_A_USER+PTE_A_WRITE+PTE_SHARED
6133                                    <1>                     ; present flag, bit 0 = 1
6134                                    <1>                     ; user flag, bit 2 = 1
6135                                    <1>                     ; writable flag, bit 1 = 1
6136                                    <1>                     ; direct memory access flag, bit 10 = 1
6137                                    <1>                     ; (This page must not be deallocated!)
6138 00006687 8902                      <1>       mov    [edx], eax  ; Update PTE value
6139 00006689 49                        <1>       dec    ecx ; remain count of contiguous pages
6140 0000668A 741E                      <1>       jz     short dmem_acc_8
6141 0000668C 81C500100000              <1>       add    ebp, PAGE_SIZE ; next physical page address
6142                                    <1>       ; 22/07/2017
6143                                    <1>       ;mov   eax, ebp
6144                                    <1>       ; 12/05/2017
6145 00006692 8B1D[AC9F0100]            <1>       mov    ebx, [base_addr] ; linear address (virtual+CORE)
6146 00006698 81C300100000              <1>       add    ebx, PAGE_SIZE      ; next linear address
6147 0000669E E938FFFFFF                <1>         jmp     dmem_acc_0
6148                                    <1> dmem_acc_7:  ; ERROR !
6149 000066A3 C7042404000000            <1>       mov    dword [esp], ERR_MINOR_IM
6150                                    <1>              ; Insufficient memory (minor) error!
6151                                    <1>              ; Major error = 0 (No protection fault)
6152                                    <1>       ; cf = 1
6153                                    <1> dmem_acc_8:
6154 000066AA 58                        <1>       pop    eax
6155                                    <1>       ;pop   edx
6156                                    <1>       ;pop   ecx
6157                                    <1>       ;pop   ebx
6158                                    <1>       ;pop   ebp
6159 000066AB C3                        <1>       retn
6160                                    <1>
6161                                    <1> deallocate_user_pages:
6162                                    <1>       ; 20/05/2017
6163                                    <1>       ; 15/05/2017
6164                                    <1>       ; 20/02/2017
6165                                    <1>       ; 19/02/2017 (TRDOS 386 = TRDOS v2.0)
6166                                    <1>       ;
6167                                    <1>       ; Deallocate virtually contiguous user pages (memory block)
6168                                    <1>       ; (caller: 'sysdalloc' system call)
6169                                    <1>       ;
6170                                    <1>       ; INPUT ->
6171                                    <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
6172                                    <1>       ;     ECX = byte count
6173                                    <1>       ;     [u.pgdir] = user's page directory
6174                                    <1>       ;     [u.ppdir] = parent's page directory
6175                                    <1>       ;
6176                                    <1>       ; OUTPUT ->
6177                                    <1>       ;    If CF = 0
6178                                    <1>       ;     EAX = Deallocated memory bytes
6179                                    <1>       ;       (Even if shared or read only pages will not be
6180                                    <1>       ;        deallocated on M.A.T., this byte count will be
6181                                    <1>       ;        returned as virtually deallocated bytes; in fact
6182                                    <1>       ;        virtually deallocated user pages * 4096.)
6183                                    <1>       ;     EBX = Virtual address (as rounded up)
6184                                    <1>       ;    If CF = 1
```

```
                                <1>       ;     EAX = 0 (there is not any deallocated pages)
                                <1>       ;
                                <1>       ; Note: Empty page tables will not be deallocated!!!
                                <1>       ;     (they will be deallocated at process termination stage)
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
                                <1>       ;
000066AC 89DE                   <1>       mov   esi, ebx
000066AE 89F7                   <1>       mov   edi, esi
000066B0 01CF                   <1>       add   edi, ecx
000066B2 81C6FF0F0000           <1>       add   esi, PAGE_SIZE - 1  ; 4095 (round up)
000066B8 C1EE0C                 <1>       shr   esi, PAGE_SHIFT
000066BB C1EF0C                 <1>       shr   edi, PAGE_SHIFT
000066BE 89F8                   <1>       mov   eax, edi ; end page
000066C0 29F0                   <1>       sub   eax, esi ; end page - start page
000066C2 0F86D5000000           <1>       jna   da_u_pd_err  ; < 1
000066C8 89F3                   <1>       mov   ebx, esi
000066CA C1E30C                 <1>       shl   ebx, PAGE_SHIFT ; virtual address (as rounded up)
000066CD 53                     <1>       push  ebx ; *
000066CE 89C1                   <1>       mov   ecx, eax ; page count
000066D0 C1E00C                 <1>       shl   eax, PAGE_SHIFT ; byte count as adjusted
000066D3 50                     <1>       push  eax ; **
000066D4 8B1D[B8030300]         <1>       mov   ebx, [u.pgdir] ; physical addr of user's page dir
000066DA 81C600040000           <1>       add   esi, CORE/PAGE_SIZE
000066E0 89F7                   <1>       mov   edi, esi
000066E2 81E7FF030000           <1>       and   edi, PTE_MASK ; PTE entry in the page table
000066E8 57                     <1>       push  edi ; *** ; PTE index (of page directory)
000066E9 C1EE0A                 <1>       shr   esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
000066EC 89F2                   <1>       mov   edx, esi
                                <1>       ; EDX = PDE index
000066EE C1E602                 <1>       shl   esi, 2 ; convert PDE index to dword offset
000066F1 01DE                   <1>       add   esi, ebx ; add page directory address
                                <1> da_u_pd_1:
000066F3 AD                     <1>       lodsd
                                <1>       ;
000066F4 89F5                   <1>       mov   ebp, esi ; 20/02/2017
                                <1>       ; EBP = next PDE address
                                <1>       ;
000066F6 A801                   <1>       test  al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
000066F8 0F8494000000           <1>       jz    da_u_pd_3 ; 20/05/2017
000066FE 662500F0               <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
                                <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
00006702 8B3C24                 <1>       mov   edi, [esp] ; ***
                                <1>       ; EDI = PTE index (of complete page directory)
                                <1>       ;and  edi, PTE_MASK ; PTE entry in the page table
00006705 C1E702                 <1>       shl   edi, 2 ; convert PTE index to dword offset
00006708 89FE                   <1>       mov   esi, edi ; PTE offset in page table (0-4092)
0000670A 01C6                   <1>       add   esi, eax ; now, esi points to requested PTE
                                <1> da_u_pt_0:
0000670C AD                     <1>       lodsd
0000670D A801                   <1>       test  al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
0000670F 743F                   <1>       jz    short da_u_pt_1
                                <1>       ;
00006711 A802                   <1>       test  al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                                <1>                               ; (must be 1)
00006713 7549                   <1>       jnz   short da_u_pt_3
                                <1>       ; Read only -duplicated- page (belongs to a parent or a child)
00006715 66A90002               <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                                <1>                                    ; as child's page ?
00006719 744E                   <1>       jz    short da_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)
0000671B 52                     <1>       push  edx ; ****
                                <1>       ; EDI = page table entry offset (0-4092)
0000671C 8B1D[BC030300]         <1>       mov   ebx, [u.ppgdir] ; page directory of the parent process
00006722 66C1E202               <1>       shl   dx, 2 ; *4
00006726 01D3                   <1>       add   ebx, edx ; PDE address (for the parent)
00006728 8B13                   <1>       mov   edx, [ebx] ; page table address
0000672A F6C201                 <1>       test  dl, PDE_A_PRESENT ; present (valid) or not ?
0000672D 742E                   <1>       jz    short da_u_pt_2     ; parent process does not use this page
0000672F 6681E200F0             <1>       and   dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
                                <1>       ; EDI = page table entry offset (0-4092)
00006734 01D7                   <1>       add   edi, edx     ; PTE address (for the parent)
00006736 8B1F                   <1>       mov   ebx, [edi]
00006738 F6C301                 <1>       test  bl, PTE_A_PRESENT ; present or not ?
0000673B 7420                   <1>       jz    short da_u_pt_2     ; parent process does not use this page
0000673D 662500F0               <1>       and   ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00006741 6681E300F0             <1>       and   bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00006746 39D8                   <1>       cmp   eax, ebx     ; parent's and child's pages are same ?
00006748 7513                   <1>       jne   short da_u_pt_2     ; not same page
                                <1>                               ; deallocate the child's page
0000674A 800F02                 <1>         or      byte [edi], PTE_A_WRITE ; convert to writable page (parent)
0000674D 5A                     <1>       pop   edx ; ****
0000674E EB19                   <1>       jmp   short da_u_pt_4
                                <1> da_u_pt_1:
00006750 09C0                   <1>       or    eax, eax     ; swapped page ?
00006752 741C                   <1>       jz    short da_u_pt_5     ; no
                                <1>                               ; yes
00006754 D1E8                   <1>       shr   eax, 1
00006756 E8D1F9FFFF             <1>       call  unlink_swap_block ; Deallocate swapped page block
                                <1>                               ; on the swap disk (or in file)
0000675B EB13                   <1>       jmp   short da_u_pt_5
                                <1> da_u_pt_2:
0000675D 5A                     <1>       pop   edx ; ****
                                <1> da_u_pt_3:
0000675E 66A90004               <1>       test  ax, PTE_SHARED      ; shared or direct memory access indicator
00006762 7505                   <1>       jnz   short da_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
00006764 E81CF5FFFF             <1>       call  deallocate_page ; set the mem allocation bit of this page
                                <1> da_u_pt_4:
00006769 C746FC00000000         <1>       mov   dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
                                <1> da_u_pt_5:
                                <1>       ; 20/05/2017
```

```
6290 00006770 58                <1>       pop    eax ; *** PTE index (of page directory)
6291 00006771 49                <1>       dec    ecx ; remain page count
6292 00006772 7426              <1>       jz     short da_u_pd_4
6293 00006774 40                <1>       inc    eax ; next PTE
6294 00006775 6625FF03          <1>       and    ax, PTE_MASK ; PTE entry index in the page table
6295 00006779 50                <1>       push   eax ; *** (save again)
6296                            <1>       ;mov   edi, eax
6297                            <1>       ;and   di, PTE_MASK
6298                            <1>       ;cmp   edi, PAGE_SIZE / 4 ; 1024
6299                            <1>       ;jnb   short da_u_pd_2
6300 0000677A 89C7              <1>       mov    edi, eax
6301 0000677C C1E702            <1>       shl    edi, 2 ; convert index to dword offset
6302                            <1>       ;test  ax, PTE_MASK ; 3FFh
6303 0000677F 09C0              <1>       or     eax, eax
6304 00006781 7589              <1>       jnz    short da_u_pt_0 ; 1-1023
6305                            <1> da_u_pd_2:
6306 00006783 42                <1>       inc    edx
6307                            <1>       ; 20/05/2017
6308 00006784 6681E2FF03        <1>       and    dx, PTE_MASK  ; 3FFh
6309 00006789 740F              <1>       jz     short da_u_pd_4  ; 0 (1024)
6310                            <1>       ;cmp   edx, 1024
6311                            <1>       ;jnb   short da_u_pd_4
6312 0000678B 89EE              <1>       mov    esi, ebp ; 20/02/2017
6313 0000678D E961FFFFFF        <1>       jmp    da_u_pd_1
6314                            <1> da_u_pd_3:
6315                            <1>       ; 15/05/2017 (empty page directory entry)
6316 00006792 81E900040000      <1>       sub    ecx, 1024
6317 00006798 77E9              <1>       ja     short da_u_pd_2 ; 20/05/2017
6318                            <1> da_u_pd_4:
6319 0000679A 58                <1>       pop    eax ; **
6320 0000679B 5B                <1>       pop    ebx ; *
6321 0000679C C3                <1>       retn
6322                            <1>
6323                            <1> da_u_pd_err:
6324 0000679D 31C0              <1>       xor    eax, eax
6325 0000679F F9                <1>       stc
6326 000067A0 C3                <1>       retn
6327                            <1>
6328                            <1> allocate_user_pages:
6329                            <1>       ; 20/05/2017
6330                            <1>       ; 01/05/2017, 02/05/2017, 15/05/2017
6331                            <1>       ; 04/03/2017
6332                            <1>       ; 20/02/2017 (TRDOS 386 = TRDOS v2.0)
6333                            <1>       ;
6334                            <1>       ; Allocate physically contiguous user pages (memory block)
6335                            <1>       ; (caller: 'sysalloc' system call)
6336                            <1>       ;
6337                            <1>       ; Note: This procedure does not alloc a page's itself
6338                            <1>       ;     (page bit) on Memory Allocation Table.
6339                            <1>       ;     (allocate_memory_block is needed before this proc)
6340                            <1>       ;
6341                            <1>       ; INPUT ->
6342                            <1>       ;     EAX = PHYSICAL ADDRESS (beginning address)
6343                            <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
6344                            <1>       ;     ECX = byte count (>=4096)
6345                            <1>       ;     [u.pgdir] = user's page directory
6346                            <1>       ;
6347                            <1>       ;     Note: All addresses are (must be) already adjusted
6348                            <1>       ;     to page      borders, otherwise, lower 12bits of addresses
6349                            <1>       ;     and byte count would be truncated.
6350                            <1>       ;
6351                            <1>       ; OUTPUT ->
6352                            <1>       ;     none
6353                            <1>       ;
6354                            <1>       ;     CF = 1 -> insufficient memory error
6355                            <1>       ;
6356                            <1>       ; Note: All pages will be allocated in physical page order
6357                            <1>       ;     from the beginning page address.
6358                            <1>       ;     * A new page table will be added to the page dir
6359                            <1>       ;       when the requested PDE is invalid.
6360                            <1>       ;     * Those pages will not be added to swap queue
6361                            <1>       ;       because main purpose of this allocation is to
6362                            <1>       ;       set a direct memory access (DMA controller) buffer.
6363                            <1>       ;      (Swapping out a page in a DMA buffer would be wrong!)
6364                            <1>       ;     * Previous content of page tables (PTEs) would be
6365                            <1>       ;       (should be) deallocated before entering this
6366                            <1>       ;       procedure. So, new page table entries (PTEs)
6367                            <1>       ;       directly will be written without checking
6368                            <1>       ;       their previous content.
6369                            <1>       ;     * Only solution to increase free memory by removing
6370                            <1>       ;       that non-swappable memory block is to terminate
6371                            <1>       ;       the process or to wait until the process will
6372                            <1>       ;       deallocate that memory block as itself. ('sysdalloc')
6373                            <1>       ;       (No problem, if the process does not grab all of
6374                            <1>       ;       -very big amount of- free memory by using
6375                            <1>       ;       'sysalloc' system call!?)
6376                            <1>       ;       (Even if the process has grabbed all of free memory,
6377                            <1>       ;       no problem if the process is not running in
6378                            <1>       ;       multitasking mode. No problem in multitasking
6379                            <1>       ;       mode if there is not another process which is running
6380                            <1>       ;       or waiting or sleeping for an event as it's pages
6381                            <1>       ;       are swapped-out. But a new process can not start to
6382                            <1>       ;       run if all of free memory has beeen allocated
6383                            <1>       ;       by running processes. Deallocation -'sysdalloc'-
6384                            <1>       ;       or terminate a running process is needed
6385                            <1>       ;       in order to run a new process.)
6386                            <1>       ;
6387                            <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
6388                            <1>       ;
6389                            <1>
6390                            <1>       ; 01/05/2017
6391 000067A1 662500F0          <1>       and    ax, ~PAGE_OFF
6392 000067A5 6681E300F0        <1>       and    bx, ~PAGE_OFF
6393                            <1>       ; 02/05/2017
6394 000067AA BD00F0FFFF        <1>       mov    ebp, 0FFFFF000h ; 4 Giga Bytes - 4096 Bytes (for Stack)
```

```
6395 000067AF C1E90C              <1>       shr    ecx, PAGE_SHIFT ; page count
6396 000067B2 83F901              <1>       cmp    ecx, 1
6397 000067B5 7251                <1>       jb     short a_u_im_retn
6398 000067B7 89C2                <1>       mov    edx, eax
6399 000067B9 01CA                <1>       add    edx, ecx
6400 000067BB 724B                <1>       jc     short a_u_im_retn
6401 000067BD 39D5                <1>       cmp    ebp, edx
6402 000067BF 7247                <1>       jb     short a_u_im_retn
6403 000067C1 89DA                <1>       mov    edx, ebx
6404 000067C3 81C200004000        <1>       add    edx, CORE
6405 000067C9 723D                <1>       jc     short a_u_im_retn
6406 000067CB 01CA                <1>       add    edx, ecx
6407 000067CD 7239                <1>       jc     short a_u_im_retn
6408 000067CF 39D5                <1>       cmp    ebp, edx
6409 000067D1 7235                <1>       jb     short a_u_im_retn
6410                              <1>       ;
6411 000067D3 89C5                <1>       mov    ebp, eax ; physical address
6412 000067D5 89DE                <1>       mov    esi, ebx
6413 000067D7 81C600004000        <1>       add    esi, CORE ; start of user's memory (4M)
6414 000067DD C1EE0C              <1>       shr    esi, PAGE_SHIFT ; higher 20 bits of the linear address
6415                              <1>       ;shr   ecx, PAGE_SHIFT ; page count
6416 000067E0 8B1D[B8030300]      <1>       mov    ebx, [u.pgdir] ; physical addr of user's page dir
6417 000067E6 89F7                <1>       mov    edi, esi
6418 000067E8 81E7FF030000        <1>       and    edi, PTE_MASK ; PTE entry index in the page table
6419 000067EE 57                  <1>       push   edi  ; * ; PTE index (in page directory)
6420 000067EF C1EE0A              <1>       shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
6421 000067F2 89F2                <1>       mov    edx, esi
6422                              <1>       ; EDX = PDE index
6423 000067F4 C1E602              <1>       shl    esi, 2 ; convert PDE index to dword offset
6424 000067F7 01DE                <1>       add    esi, ebx ; add page directory address
6425                              <1> a_u_pd_0:
6426 000067F9 AD                  <1>       lodsd
6427                              <1>       ;
6428 000067FA 89F3                <1>       mov    ebx, esi ; next PDE address
6429                              <1>       ;
6430 000067FC A801                <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
6431 000067FE 7513                <1>       jnz    short a_u_pd_2
6432                              <1>       ;
6433                              <1>       ; empty PDE (it does not point to valid page table address)
6434 00006800 E8A2F2FFFF          <1>       call   allocate_page  ; (allocate a new page table)
6435 00006805 7302                <1>       jnc    short a_u_pd_1 ; OK... now, we have a new page table.
6436                              <1>       ; cf = 1
6437                              <1>       ; There is not a free memory page to allocate a new page table !!!
6438 00006807 5E                  <1>       pop    esi ; *
6439                              <1> a_u_im_retn:
6440 00006808 C3                  <1>       retn   ; return to 'sysalloc' with 'insufficient memory' error
6441                              <1>       ;
6442                              <1> a_u_pd_1: ; clear the new page table content
6443                              <1>       ; EAX = Physical (base) address of the new page table
6444 00006809 E813F3FFFF          <1>       call   clear_page ; Clear page content
6445                              <1>       ;
6446 0000680E 0C07                <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
6447                              <1>              ; set bit 0, bit 1 and bit 2 to 1
6448                              <1>              ; (present, writable, user)
6449 00006810 8946FC              <1>       mov    [esi-4], eax
6450                              <1> a_u_pd_2:
6451 00006813 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
6452                              <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
6453 00006817 8B3C24              <1>       mov    edi, [esp] ; *
6454                              <1>       ; EDI = PTE index (of page directory)
6455                              <1>       ;and   edi, PTE_MASK ; PTE entry index in the page table
6456                              <1>       ; EBX = next PDE address
6457 0000681A 89FE                <1>       mov    esi, edi ; PTE index in page table (0-1023)
6458 0000681C C1E702              <1>       shl    edi, 2 ; convert PTE index to dword offset
6459 0000681F 01C7                <1>       add    edi, eax ; now, edi points to requested PTE
6460                              <1> a_u_pt_0:
6461                              <1>       ; 02/05/2017
6462 00006821 8B07                <1>       mov    eax, [edi]
6463                              <1>       ;
6464 00006823 A801                <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
6465 00006825 7445                <1>       jz     short a_u_pt_1
6466                              <1>       ;
6467 00006827 A802                <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
6468                              <1>                                ; (must be 1)
6469 00006829 7550                <1>       jnz    short a_u_pt_3
6470                              <1>       ; Read only -duplicated- page (belongs to a parent or a child)
6471 0000682B 66A90002            <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
6472                              <1>                                    ; as child's page ?
6473 0000682F 7455                <1>       jz     short a_u_pt_4      ; Clear PTE but don't deallocate the page!
6474                              <1>       ;
6475                              <1>       ; check the parent's PTE value is read only & same page or not..
6476                              <1>       ; EDX = page directory entry index (0-1023)
6477 00006831 52                  <1>       push   edx ; **
6478 00006832 53                  <1>       push   ebx ; ***
6479                              <1>       ; ESI = page table entry index (0-1023)
6480                              <1>       ;push  esi ; **** ; 20/05/2017
6481 00006833 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
6482 00006839 66C1E202            <1>       shl    dx, 2 ; *4
6483 0000683D 01D3                <1>       add    ebx, edx ; PTE address,0 (for the parent)
6484 0000683F 8B13                <1>       mov    edx, [ebx] ; page table address
6485 00006841 F6C201              <1>       test   dl, PDE_A_PRESENT ; present (valid) or not ?
6486 00006844 7433                <1>       jz     short a_u_pt_2      ; parent process does not use this page
6487 00006846 6681E200F0          <1>       and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
6488 0000684B 66C1E602            <1>       shl    si, 2 ; *4
6489                              <1>       ; ESI = page table entry offset (0-4092)
6490 0000684F 01D6                <1>       add    esi, edx     ; PTE address (for the parent)
6491 00006851 8B1E                <1>       mov    ebx, [esi]
6492 00006853 F6C301              <1>       test   bl, PTE_A_PRESENT ; present or not ?
6493 00006856 7421                <1>       jz     short a_u_pt_2      ; parent process does not use this page
6494 00006858 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6495 0000685C 6681E300F0          <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6496 00006861 39D8                <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
6497 00006863 7514                <1>       jne    short a_u_pt_2      ; not same page
6498                              <1>                                ; deallocate the child's page
6499 00006865 800E02              <1>         or     byte [esi], PTE_A_WRITE ; convert to writable page (parent)
```

```
6500                                    <1>       ;pop   esi ; **** ; 20/05/2017
6501 00006868 5B                        <1>       pop    ebx ; ***
6502 00006869 5A                        <1>       pop    edx ; **
6503 0000686A EB1A                      <1>       jmp    short a_u_pt_4
6504                                    <1> a_u_pt_1:
6505 0000686C 09C0                      <1>       or     eax, eax     ; swapped page ?
6506 0000686E 7416                      <1>       jz     short a_u_pt_4      ; no
6507                                    <1>                                  ; yes
6508 00006870 D1E8                      <1>       shr    eax, 1
6509 00006872 E8B5F8FFFF                <1>       call   unlink_swap_block ; Deallocate swapped page block
6510                                    <1>                                ; on the swap disk (or in file)
6511 00006877 EB0D                      <1>       jmp    short a_u_pt_4
6512                                    <1> a_u_pt_2:
6513                                    <1>       ;pop   esi ; **** ; 20/05/2017
6514 00006879 5B                        <1>       pop    ebx ; ***
6515 0000687A 5A                        <1>       pop    edx ; **
6516                                    <1> a_u_pt_3:
6517 0000687B 66A90004                  <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
6518 0000687F 7505                      <1>       jnz    short a_u_pt_4      ; AVL bit 1 = 1, do not deallocate this page!
6519                                    <1>       ;
6520                                    <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
6521 00006881 E8FFF3FFFF                <1>       call   deallocate_page ; set the mem allocation bit of this page
6522                                    <1>       ;
6523                                    <1> a_u_pt_4:
6524 00006886 89E8                      <1>       mov    eax, ebp ; physical address
6525 00006888 0C07                      <1>       or     al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 04/03/2017
6526 0000688A AB                        <1>       stosd
6527 0000688B 5E                        <1>       pop    esi ; * ; 20/05/2017
6528 0000688C 49                        <1>       dec    ecx ; remain page count
6529 0000688D 7417                      <1>       jz     short a_u_pd_5
6530 0000688F 81C500100000              <1>       add    ebp, PAGE_SIZE
6531 00006895 46                        <1>       inc    esi ; next PTE (index)
6532                                    <1>       ; 20/05/2017
6533                                    <1>       ;cmp   esi, PAGE_SIZE/4 ; 1024
6534                                    <1>       ;jb    short a_u_pt_0
6535 00006896 6681E6FF03                <1>       and    si, PTE_MASK ; 3FFh (0 to 1023)
6536 0000689B 56                        <1>       push   esi ; *
6537 0000689C 7583                      <1>       jnz    short a_u_pt_0 ; > 0 (<1024)
6538                                    <1> a_u_pd_3:
6539 0000689E 42                        <1>       inc    edx
6540                                    <1> ;     cmp    edx, 1024
6541                                    <1> ;     jnb    short a_u_pd_4 ; 02/05/2017 (error!, ecx > 0)
6542 0000689F 89DE                      <1>       mov    esi, ebx ; the next PDE address
6543 000068A1 E953FFFFFF                <1>       jmp    a_u_pd_0
6544                                    <1> a_u_pd_4:
6545                                    <1>       ; 02/05/2017
6546                                    <1> ;     stc
6547                                    <1> a_u_pd_5:
6548                                    <1>       ; 20/05/2017
6549                                    <1>       ;pop   edi ; *
6550 000068A6 C3                        <1>       retn
6551                                    <1>
6552                                    <1> allocate_lfb_pages_for_kernel:
6553                                    <1>       ; 15/12/2020
6554                                    <1>       ; 14/12/2020 - TRDOS 386 v2.0.3
6555                                    <1>       ; Set kernel page tables for linear frame buffer
6556                                    <1>       ; (this procedure will be called by kernel only)
6557                                    <1>       ;
6558                                    <1>       ; Input:
6559                                    <1>       ;     [LFB_ADDR] = linear frame buffer base address
6560                                    <1>       ;     [LFB_SIZE] = linear frame buffer size in bytes
6561                                    <1>       ; Output:
6562                                    <1>       ;     none
6563                                    <1>       ;     cf = 1 -> error
6564                                    <1>       ;
6565                                    <1>       ; Modified registers: eax, ecx, edx, edi
6566                                    <1>
6567 000068A7 8B3D[2C120300]            <1>       mov    edi, [LFB_ADDR]
6568 000068AD 8B15[30120300]            <1>       mov    edx, [LFB_SIZE]
6569                                    <1>
6570 000068B3 C1EF16                    <1>       shr    edi, 22      ; convert address to page number
6571                                    <1>                         ; and then convert it to PDE entry offset
6572                                    <1>                         ; (1 PDE is for 4MB, 22 bit shift)
6573                                    <1>
6574 000068B6 66C1E702                  <1>       shl    di, 2  ; * 4 for offset
6575                                    <1>
6576                                    <1>       ;add   edx, 4095
6577 000068BA C1EA0C                    <1>       shr    edx, 12      ; convert LFB size to LFB page count
6578                                    <1>
6579 000068BD 89D1                      <1>       mov    ecx, edx ; * ; LFB page count
6580                                    <1>
6581 000068BF 81C1FF030000              <1>       add    ecx, 1023 ; page count + 1023
6582 000068C5 C1E90A                    <1>       shr    ecx, 10 ; convert to page directory entry count
6583                                    <1>                      ; (page table count)
6584 000068C8 51                        <1>       push   ecx ; **
6585 000068C9 C1E10C                    <1>       shl    ecx, 12 ; convert to byte count
6586                                    <1>
6587 000068CC 31C0                      <1>       xor    eax, eax ; first available pages
6588                                    <1>
6589                                    <1>       ; allocate contiguous memory block for these kernel pages
6590                                    <1>
6591 000068CE E87EFAFFFF                <1>       call   allocate_memory_block
6592                                    <1>       ; eax = start address of (contiguous) memory block
6593 000068D3 59                        <1>       pop    ecx ; ** ; PDE count
6594 000068D4 7301                      <1>       jnc    short a_lfb_k_1
6595                                    <1>       ; error (cf=1)
6596 000068D6 C3                        <1>       retn
6597                                    <1> a_lfb_k_1:
6598                                    <1>       ; Allocate (new) page tables in kernel's page directory
6599 000068D7 51                        <1>       push   ecx ; PDE (page table) count
6600 000068D8 50                        <1>       push   eax ; start address of contiguous memory pages
6601                                    <1>                  ; (at page boundary)
6602                                    <1>       ; edi = 1st page directory entry offset
6603 000068D9 033D[F0880100]            <1>       add    edi, [k_page_dir] ; Kernel's Page Dir Address
6604                                    <1> a_lfb_k_2:
```

```
6605 000068DF 660D0304            <1>       or    ax, PDE_A_PRESENT + PDE_A_WRITE + PDE_EXTERNAL
6606                              <1>                             ; supervisor + read&write + present
6607                              <1>                             ; + external memory block (LFB)
6608 000068E3 AB                  <1>       stosd
6609 000068E4 0500100000          <1>       add   eax, 4096
6610 000068E9 E2F4                <1>       loop  a_lfb_k_2
6611                              <1>
6612 000068EB 5F                  <1>       pop   edi ; start addr of contiguous memory pages
6613 000068EC 59                  <1>       pop   ecx ; page table (PDE) count
6614                              <1>
6615                              <1>       ; Allocate pages in (new) kernel page tables
6616                              <1>
6617                              <1>       ; (Note: page tables are contiguous in pyhsical memory)
6618 000068ED C1E10A              <1>       shl   ecx, 10 ; * 1024, convert to (total) PTE count
6619                              <1>
6620 000068F0 A1[2C120300]        <1>       mov   eax, [LFB_ADDR]
6621                              <1>       ; edx = LFB page count
6622                              <1>       ;and  ax, ~4095  ; lw of LFB address is 0
6623                              <1> a_lfb_k_3:
6624 000068F5 660D0304            <1>       or    ax, PTE_A_PRESENT + PTE_A_WRITE + PTE_EXTERNAL
6625                              <1>                             ; supervisor + read&write + present
6626                              <1>                             ; + external memory block (LFB)
6627 000068F9 AB                  <1>       stosd
6628 000068FA 4A                  <1>       dec   edx
6629 000068FB 7408                <1>       jz    short a_lfb_k_4 ; LFB size has been completed (!?)
6630 000068FD 0500100000          <1>       add   eax, 4096
6631 00006902 E2F1                <1>       loop  a_lfb_k_3
6632                              <1>
6633 00006904 C3                  <1>       retn
6634                              <1>
6635                              <1> a_lfb_k_4:
6636                              <1>       ; clear PTEs for empty/free pages
6637                              <1>       ;     (if there are after LFB !?)
6638 00006905 31C0                <1>       xor   eax, eax ; clear page table entry (empty)
6639 00006907 F3AB                <1>       rep   stosd
6640 00006909 C3                  <1>       retn
6641                              <1>
6642                              <1> ;deallocate_lfb_pages_for_kernel:
6643                              <1>       ; 15/12/2020
6644                              <1>       ; 14/12/2020 - TRDOS 386 v2.0.3
6645                              <1>       ; Reset/Release kernel page tables
6646                              <1>       ;      which are used for linear frame buffer
6647                              <1>       ; (this procedure will be called by kernel only)
6648                              <1>       ;
6649                              <1>       ; Input:
6650                              <1>       ;     [LFB_ADDR] = linear frame buffer base address
6651                              <1>       ;     [FFB_SIZE] = linear frame buffer size in bytes
6652                              <1>       ; Output:
6653                              <1>       ;     none
6654                              <1>       ;
6655                              <1>       ; Modified registers: eax, ecx, edi
6656                              <1>
6657                              <1>       ;mov   edi, [LFB_ADDR]
6658                              <1>       ;mov   ecx, [LFB_SIZE]
6659                              <1>       ;
6660                              <1>       ;shr   edi, 22      ; convert address to page number
6661                              <1>       ;            ; and then convert it to PDE entry offset
6662                              <1>       ;            ; (1 PDE is for 4MB, 22 bit shift)
6663                              <1>       ;
6664                              <1>       ;shl   di, 2  ; * 4 for offset
6665                              <1>       ;
6666                              <1>       ;;add  ecx, 4095
6667                              <1>       ;shr   ecx, 12      ; convert LFB size to page count
6668                              <1>       ;
6669                              <1>       ;add   ecx, 1023 ; page count + 1023
6670                              <1>       ;shr   ecx, 10 ; convert to page directory entry count
6671                              <1>       ;            ; (page table count)
6672                              <1>       ;push  ecx ; *
6673                              <1>       ;shl   ecx, 12 ; convert to byte count
6674                              <1>       ;
6675                              <1>       ;xor   eax, eax ; first available pages
6676                              <1>       ;
6677                              <1>       ;; deallocate contiguous memory block for kernel pages
6678                              <1>       ;
6679                              <1>       ;call  deallocate_memory_block
6680                              <1>       ;
6681                              <1>       ;pop   ecx ; * ; PDE count
6682                              <1>       ;
6683                              <1>       ;; Release/Free PDEs (page tables) in kernel's page dir
6684                              <1>       ;; edi = 1st page directory entry offset
6685                              <1>       ;add   edi, [k_page_dir] ; Kernel's Page Dir Address
6686                              <1>       ;sub   eax, eax ; clear (also invalidate)
6687                              <1>       ;rep   stosd
6688                              <1>       ;
6689                              <1>       ;retn
6690                              <1>
6691                              <1> ; /// End Of MEMORY MANAGEMENT FUNCTIONS ///
6692                              <1>
6693                              <1> ;; Data:
6694                              <1>
6695                              <1> ; 09/03/2015
6696                              <1> ;swpq_count: dw 0 ; count of pages on the swap que
6697                              <1> ;swp_drv:    dd 0 ; logical drive description table address of the swap drive/disk
6698                              <1> ;swpd_size:  dd 0 ; size of swap drive/disk (volume) in sectors (512 bytes).

6699                              <1> ;swpd_free:  dd 0 ; free page blocks (4096 bytes) on swap disk/drive (logical)
6700                              <1> ;swpd_next:  dd 0 ; next free page block
6701                              <1> ;swpd_last:  dd 0 ; last swap page block
2893                                  %include 'timer.s'   ; 17/01/2015
2894                              <1> ; ****************************************************************************
2895                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - timer.s
2896                              <1> ; ----------------------------------------------------------------------------
2897                              <1> ; Last Update: 15/01/2017
2898                              <1> ; ----------------------------------------------------------------------------
2899                              <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> ;
                                <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
                                <1> ; ****************************************************************************
                                <1>
                                <1> ; TRDOS 386  (TRDOS v2.0) Kernel - TIMER & REAL TIME CLOCK (BIOS) FUNCTIONS
                                <1>
                                <1> ; IBM PC-AT BIOS Source Code ('BIOS2.ASM')
                                <1> ; TITLE BIOS2 ---- 06/10/85 BIOS INTERRUPT ROUTINES
                                <1>
                                <1> ;
                                <1> ; ///////// TIMER (& REAL TIME CLOCK) FUNCTIONS ///////////////
                                <1>
                                <1> int1Ah:
                                <1>       ; 29/01/2016
                                <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
0000690A 9C                     <1>       pushfd
0000690B 0E                     <1>       push   cs
0000690C E801000000             <1>       call   TIME_OF_DAY_1
00006911 C3                     <1>       retn
                                <1>
                                <1> ;--- INT  1A H -- (TIME OF DAY) ------------------------------------------------
                                <1> ;       THIS BIOS ROUTINE ALLOWS THE CLOCKS TO BE SET OR READ                 :
                                <1> ;                                                                             :
                                <1> ; PARAMETERS:                                                                 :
                                <1> ;     (AH) = 00H  READ THE CURRENT SETTING AND RETURN WITH,                   :
                                <1> ;                       (CX) = HIGH PORTION OF COUNT                          :
                                <1> ;                       (DX) = LOW PORTION OF COUNT                           :
                                <1> ;                       (AL) = 0 TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ :
                                <1> ;                              1 IF ON ANOTHER DAY. (RESET TO ZERO AFTER READ) :
                                <1> ;                                                                             :
                                <1> ;     (AH) = 01H  SET THE CURRENT CLOCK USING,                                :
                                <1> ;                 (CX) = HIGH PORTION OF COUNT                                :
                                <1> ;                 (DX) = LOW PORTION OF COUNT.                                :
                                <1> ;                                                                             :
                                <1> ;               NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SECOND :
                                <1> ;                             (OR ABOUT 18.2 PER SECOND -- SEE EQUATES)       :
                                <1> ;                                                                             :
                                <1> ;     (AH) = 02H  READ THE REAL TIME CLOCK AND RETURN WITH,                   :
                                <1> ;                       (CH) = HOURS IN BCD (00-23)                           :
                                <1> ;                       (CL) = MINUTES IN BCD (00-59)                         :
                                <1> ;                       (DH) = SECONDS IN BCD (00-59)                         :
                                <1> ;                       (DL) = DAYLIGHT SAVINGS ENABLE (00-01)                :
                                <1> ;                                                                             :
                                <1> ;     (AH) = 03H  SET THE REAL TIME CLOCK USING,                              :
                                <1> ;                     (CH) = HOURS IN BCD (00-23)                             :
                                <1> ;                     (CL) = MINUTES IN BCD (00-59)                           :
                                <1> ;                     (DH) = SECONDS IN BCD (00-59)                           :
                                <1> ;                     (DL) = 01 IF DAYLIGHT SAVINGS ENABLE OPTION, ELSE 00.   :
                                <1> ;                                                                             :
                                <1> ;              NOTE: (DL) = 00 IF DAYLIGHT SAVINGS TIME ENABLE IS NOT ENABLED. :
                                <1> ;                    (DL) = 01 ENABLES TWO SPECIAL UPDATES THE LAST SUNDAY IN  :
                                <1> ;                  APRIL   (1:59:59 --> 3:00:00 AM) AND THE LAST SUNDAY IN    :
                                <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
```

```
3005 00006912 80642408FE          <1>       and    byte [esp+8], 11111110b    ; clear carry bit of eflags register
3006                              <1>       ;
3007 00006917 80FC08              <1>       cmp    ah, (RTC_TBE-RTC_TB)/4     ; CHECK IF COMMAND IN VALID RANGE (0-7)
3008 0000691A F5                  <1>       cmc                           ; COMPLEMENT CARRY FOR ERROR EXIT
3009                              <1>       ; (*) jc short TIME_9          ; EXIT WITH CARRY = 1 IF NOT VALID
3010 0000691B 721A                <1>       jc     short _TIME_9 ; 29/05/2016
3011                              <1>
3012 0000691D 1E                  <1>       push   ds
3013 0000691E 56                  <1>       push   esi
3014 0000691F 66BE1000            <1>       mov    si, KDATA              ; kernel data segment
3015 00006923 8EDE                <1>       mov    ds, si
3016                              <1>
3017                              <1>       ;;15/01/2017
3018                              <1>       ; 14/01/2017
3019                              <1>       ; 02/01/2017
3020                              <1>       ;;mov  byte [intflg], 35h  ; date & time interrupt
3021                              <1>       ;sti
3022                              <1>       ;
3023 00006925 C0E402              <1>       shl    ah, 2                  ; convert function to dword offset
3024 00006928 0FB6F4              <1>       movzx  esi, ah                        ; PLACE INTO ADDRESSING REGISTER
3025                              <1>       ;cli                           ; NO INTERRUPTS DURING TIME FUNCTIONS
3026 0000692B FF96[3D690000]      <1>       call   [esi+RTC_TB]            ; VECTOR TO FUNCTION REQUESTED WITH CY=0
3027                              <1>                                      ; RETURN WITH CARRY FLAG SET FOR RESULT
3028                              <1>       ;sti                           ; INTERRUPTS BACK ON
3029 00006931 B400                <1>       mov    ah, 0                  ; CLEAR (AH) TO ZERO
3030 00006933 5E                  <1>       pop    esi                    ; RECOVER USERS REGISTER
3031 00006934 1F                  <1>       pop    ds                     ; RECOVER USERS SEGMENT SELECTOR
3032                              <1>
3033                              <1>       ;;15/01/2017
3034                              <1>       ; 02/01/2017
3035                              <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
3036                              <1>
3037                              <1> ;TIME_9:
3038                              <1>                                      ; RETURN WITH CY= 0 IF NO ERROR
3039                              <1>       ; (*) 29/05/2016
3040                              <1>       ; (*) retf 4 ; skip eflags on stack
3041 00006935 7305                <1>       jnc    short _TIME_10
3042                              <1> _TIME_9:
3043                              <1>       ; 29/05/2016 -set carry flag on stack-
3044                              <1>       ; [esp] = EIP
3045                              <1>       ; [esp+4] = CS
3046                              <1>       ; [esp+8] = E-FLAGS
3047 00006937 804C240801          <1>       or     byte [esp+8], 1      ; set carry bit of eflags register
3048                              <1>       ; [esp+12] = ESP (user)
3049                              <1>       ; [esp+16] = SS (User)
3050                              <1> _TIME_10:
3051 0000693C CF                  <1>       iretd
3052                              <1>
3053                              <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
3054                              <1>       ; (OUTER-PRIVILEGE-LEVEL)
3055                              <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
3056                              <1>       ; // RETF instruction:
3057                              <1>       ;
3058                              <1>       ; IF OperandMode=32 THEN
3059                              <1>       ;    Load CS:EIP from stack;
3060                              <1>       ;    Set CS RPL to CPL;
3061                              <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
3062                              <1>       ;    Load SS:eSP from stack;
3063                              <1>       ; ELSE (* OperandMode=16 *)
3064                              <1>       ;    Load CS:IP from stack;
3065                              <1>       ;    Set CS RPL to CPL;
3066                              <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
3067                              <1>       ;    Load SS:eSP from stack;
3068                              <1>       ; FI;
3069                              <1>       ;
3070                              <1>       ; //
3071                              <1>                                      ; ROUTINE VECTOR TABLE (AH)=
3072                              <1> RTC_TB:
3073 0000693D [5D690000]          <1>       dd     RTC_00                  ; 0 = READ CURRENT CLOCK COUNT
3074 00006941 [70690000]          <1>       dd     RTC_10                  ; 1 = SET CLOCK COUNT
3075 00006945 [7E690000]          <1>       dd     RTC_20                  ; 2 = READ THE REAL TIME CLOCK TIME
3076 00006949 [AD690000]          <1>       dd     RTC_30                  ; 3 = SET REAL TIME CLOCK TIME
3077 0000694D [EF690000]          <1>       dd     RTC_40                  ; 4 = READ THE REAL TIME CLOCK DATE
3078 00006951 [1C6A0000]          <1>       dd     RTC_50                  ; 5 = SET REAL TIME CLOCK DATE
3079 00006955 [696A0000]          <1>       dd     RTC_60                  ; 6 = SET THE REAL TIME CLOCK ALARM
3080 00006959 [BC6A0000]          <1>       dd     RTC_70                  ; 7 = RESET ALARM
3081                              <1>
3082                              <1> RTC_TBE      equ    $
3083                              <1>
3084                              <1> RTC_00:                               ; READ TIME COUNT
3085 0000695D A0[74890100]        <1>       mov    al, [TIMER_OFL]               ; GET THE OVERFLOW FLAG
3086 00006962 C605[74890100]00    <1>       mov    byte [TIMER_OFL], 0 ; AND THEN RESET THE OVERFLOW FLAG
3087 00006969 8B0D[70890100]      <1>         mov     ecx, [TIMER_LH]          ; GET COUNT OF TIME
3088 0000696F C3                  <1>       retn
3089                              <1>
3090                              <1> RTC_10:                               ; SET TIME COUNT
3091 00006970 890D[70890100]      <1>         mov     [TIMER_LH], ecx          ; SET TIME COUNT
3092 00006976 C605[74890100]00    <1>       mov    byte [TIMER_OFL], 0 ; RESET OVERFLOW FLAG
3093 0000697D C3                  <1>       retn                           ; RETURN WITH NO CARRY
3094                              <1>
3095                              <1> RTC_20:                               ; GET RTC TIME
3096 0000697E E8EB010000          <1>       call   UPD_IPR                        ; CHECK FOR UPDATE IN PROCESS
3097 00006983 7227                <1>       jc     short RTC_29            ; EXIT IF ERROR (CY= 1)
3098                              <1>
3099 00006985 B000                <1>       mov    al, CMOS_SECONDS        ; SET ADDRESS OF SECONDS
3100 00006987 E8FD010000          <1>       call   CMOS_READ               ; GET SECONDS
3101 0000698C 88C6                <1>       mov    dh, al                  ; SAVE
3102 0000698E B00B                <1>       mov    al, CMOS_REG_B                ; ADDRESS ALARM REGISTER
3103 00006990 E8F4010000          <1>       call   CMOS_READ               ; READ CURRENT VALUE OF DSE BIT
3104 00006995 2401                <1>       and    al, 00000001b           ; MASK FOR VALID DSE BIT
3105 00006997 88C2                <1>       mov    dl, al                  ; SET [DL] TO ZERO FOR NO DSE BIT
3106 00006999 B002                <1>       mov    al, CMOS_MINUTES        ; SET ADDRESS OF MINUTES
3107 0000699B E8E9010000          <1>       call   CMOS_READ               ; GET MINUTES
3108 000069A0 88C1                <1>       mov    cl, al                  ; SAVE
3109 000069A2 B004                <1>         mov     al, CMOS_HOURS          ; SET ADDRESS OF HOURS
```

```
3110 000069A4 E8E0010000          <1>       call   CMOS_READ           ; GET HOURS
3111 000069A9 88C5                <1>       mov    ch, al              ; SAVE
3112 000069AB F8                  <1>       clc                        ; SET CY= 0
3113                              <1> RTC_29:
3114 000069AC C3                  <1>       retn                       ; RETURN WITH RESULT IN CARRY FLAG
3115                              <1>
3116                              <1> RTC_30:                          ; SET RTC TIME
3117 000069AD E8BC010000          <1>       call   UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
3118 000069B2 7305                <1>       jnc    short RTC_35        ; GO AROUND IF CLOCK OPERATING
3119 000069B4 E817010000          <1>       call   RTC_STA                   ; ELSE TRY INITIALIZING CLOCK
3120                              <1> RTC_35:
3121 000069B9 88F4                <1>       mov    ah, dh              ; GET TIME BYTE - SECONDS
3122 000069BB B000                <1>       mov    al, CMOS_SECONDS    ; ADDRESS SECONDS
3123 000069BD E8E0010000          <1>       call   CMOS_WRITE          ; UPDATE SECONDS
3124 000069C2 88CC                <1>       mov    ah, cl              ; GET TIME BYTE - MINUTES
3125 000069C4 B002                <1>       mov    al, CMOS_MINUTES    ; ADDRESS MINUTES
3126 000069C6 E8D7010000          <1>       call   CMOS_WRITE          ; UPDATE MINUTES
3127 000069CB 88EC                <1>       mov    ah, ch              ; GET TIME BYTE - HOURS
3128 000069CD B004                <1>       mov    al, CMOS_HOURS           ; ADDRESS HOURS
3129 000069CF E8CE010000          <1>       call   CMOS_WRITE          ; UPDATE ADDRESS
3130                              <1>       ;mov   al, CMOS_REG_B           ; ADDRESS ALARM REGISTER
3131                              <1>       ;mov   ah, al
3132 000069D4 66B80B0B            <1>       mov    ax, CMOS_REG_B * 257
3133 000069D8 E8AC010000          <1>       call   CMOS_READ           ; READ CURRENT TIME
3134 000069DD 2462                <1>       and    al, 01100010b       ; MASK FOR VALID BIT POSITIONS
3135 000069DF 0C02                <1>       or     al, 00000010b       ; TURN ON 24 HOUR MODE
3136 000069E1 80E201              <1>       and    dl, 00000001b       ; USE ONLY THE DSE BIT
3137 000069E4 08D0                <1>       or     al, dl              ; GET DAY LIGHT SAVINGS TIME BIT (OSE)
3138 000069E6 86E0                <1>       xchg   ah, al              ; PLACE IN WORK REGISTER AND GET ADDRESS
3139 000069E8 E8B5010000          <1>       call   CMOS_WRITE          ; SET NEW ALARM SITS
3140 000069ED F8                  <1>       clc                        ; SET CY= 0
3141 000069EE C3                  <1>       retn                       ; RETURN WITH CY= 0
3142                              <1>
3143                              <1> RTC_40:                          ; GET RTC DATE
3144 000069EF E87A010000          <1>       call   UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
3145 000069F4 7225                <1>       jc     short RTC_49        ; EXIT IF ERROR (CY= 1)
3146                              <1>
3147 000069F6 B007                <1>       mov    al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH
3148 000069F8 E88C010000          <1>       call   CMOS_READ           ; READ DAY OF MONTH
3149 000069FD 88C2                <1>       mov    dl, al              ; SAVE
3150 000069FF B008                <1>       mov    al, CMOS_MONTH           ; ADDRESS MONTH
3151 00006A01 E883010000          <1>       call   CMOS_READ           ; READ MONTH
3152 00006A06 88C6                <1>       mov    dh, al              ; SAVE
3153 00006A08 B009                <1>       mov    al, CMOS_YEAR       ; ADDRESS YEAR
3154 00006A0A E87A010000          <1>       call   CMOS_READ           ; READ YEAR
3155 00006A0F 88C1                <1>       mov    cl, al              ; SAVE
3156 00006A11 B032                <1>       mov    al, CMOS_CENTURY    ; ADDRESS CENTURY LOCATION
3157 00006A13 E871010000          <1>       call   CMOS_READ           ; GET CENTURY BYTE
3158 00006A18 88C5                <1>       mov    ch, al              ; SAVE
3159 00006A1A F8                  <1>       clc                        ; SET CY=0
3160                              <1> RTC_49:
3161 00006A1B C3                  <1>       retn                       ; RETURN WITH RESULTS IN CARRY FLAG
3162                              <1>
3163                              <1> RTC_50:                          ; SET RTC DATE
3164 00006A1C E84D010000          <1>       call   UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
3165 00006A21 7305                <1>       jnc    short RTC_55        ; GO AROUND IF NO ERROR
3166 00006A23 E8A8000000          <1>       call   RTC_STA                   ; ELSE INITIALIZE CLOCK
3167                              <1> RTC_55:
3168 00006A28 66B80600            <1>       mov    ax, CMOS_DAY_WEEK   ; ADDRESS OF DAY OF WEEK BYTE
3169 00006A2C E871010000          <1>       call   CMOS_WRITE          ; LOAD ZEROS TO DAY OF WEEK
3170 00006A31 88D4                <1>       mov    ah, dl              ; GET DAY OF MONTH BYTE
3171 00006A33 B007                <1>       mov    al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH BYTE
3172 00006A35 E868010000          <1>       call   CMOS_WRITE          ; WRITE OF DAY OF MONTH REGISTER
3173 00006A3A 88F4                <1>       mov    ah, dh              ; GET MONTH
3174 00006A3C B008                <1>       mov    al, CMOS_MONTH           ; ADDRESS MONTH BYTE
3175 00006A3E E85F010000          <1>       call   CMOS_WRITE          ; WRITE MONTH REGISTER
3176 00006A43 88CC                <1>       mov    ah, cl              ; GET YEAR BYTE
3177 00006A45 B009                <1>       mov    al, CMOS_YEAR       ; ADDRESS YEAR REGISTER
3178 00006A47 E856010000          <1>       call   CMOS_WRITE          ; WRITE YEAR REGISTER
3179 00006A4C 88EC                <1>       mov    ah, ch              ; GET CENTURY BYTE
3180 00006A4E B032                <1>       mov    al, CMOS_CENTURY    ; ADDRESS CENTURY BYTE
3181 00006A50 E84D010000          <1>       call   CMOS_WRITE          ; WRITE CENTURY LOCATION
3182                              <1>       ;mov   al, CMOS_REG_B           ; ADDRESS ALARM REGISTER
3183                              <1>       ;mov   ah, al
3184 00006A55 66B80B0B            <1>       mov    ax, CMOS_REG_B * 257
3185 00006A59 E82B010000          <1>       call   CMOS_READ           ; READ WIRRENT SETTINGS
3186 00006A5E 247F                <1>       and    al, 07Fh            ; CLEAR 'SET BIT'
3187 00006A60 86E0                <1>       xchg   ah, al              ; MOVE TO WORK REGISTER
3188 00006A62 E83B010000          <1>       call   CMOS_WRITE          ; AND START CLOCK UPDATING
3189 00006A67 F8                  <1>       clc                        ; SET CY= 0
3190 00006A68 C3                  <1>       retn                       ; RETURN CY=0
3191                              <1>
3192                              <1> RTC_60:                          ; SET RTC ALARM
3193 00006A69 B00B                <1>       mov    al, CMOS_REG_B           ; ADDRESS ALARM
3194 00006A6B E819010000          <1>       call   CMOS_READ           ; READ ALARM REGISTER
3195 00006A70 A820                <1>       test   al, 20h                  ; CHECK FOR ALARM ALREADY ENABLED
3196 00006A72 F9                  <1>       stc                        ; SET CARRY IN CASE OF ERROR
3197 00006A73 7542                <1>       jnz    short RTC_69        ; ERROR EXIT IF ALARM SET
3198 00006A75 E8F4000000          <1>       call   UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
3199 00006A7A 7305                <1>       jnc    short RTC_65        ; SKIP INITIALIZATION IF NO ERROR
3200 00006A7C E84F000000          <1>       call   RTC_STA                   ; ELSE INITIALIZE CLOCK
3201                              <1> RTC_65:
3202 00006A81 88F4                <1>       mov    ah, dh              ; GET SECONDS BYTE
3203 00006A83 B001                <1>       mov    al, CMOS_SEC_ALARM  ; ADDRESS THE SECONDS ALARM REGISTER
3204 00006A85 E818010000          <1>       call   CMOS_WRITE          ; INSERT SECONDS
3205 00006A8A 88CC                <1>       mov    ah, cl              ; GET MINUTES PARAMETER
3206 00006A8C B003                <1>       mov    al, CMOS_MIN_ALARM  ; ADDRESS MINUTES ALARM REGISTER
3207 00006A8E E80F010000          <1>       call   CMOS_WRITE          ; INSERT MINUTES
3208 00006A93 88EC                <1>       mov    ah, ch              ; GET HOURS PARAMETER
3209 00006A95 B005                <1>       mov    al, CMOS_HR_ALARM   ; ADDRESS HOUR ALARM REGISTER
3210 00006A97 E806010000          <1>       call   CMOS_WRITE          ; INSERT HOURS
3211 00006A9C E4A1                <1>       in     al, INTB01          ; READ SECOND INTERRUPT MASK REGISTER
3212 00006A9E 24FE                <1>       and    al, 0FEh            ; ENABLE ALARM TIMER BIT (CY= 0)
3213 00006AA0 E6A1                <1>       out    INTB01, al          ; WRITE UPDATED MASK
3214                              <1>       ;mov   al, CMOS_REG_B           ; ADDRESS ALARM REGISTER
```

```
3215                            <1>       ;mov   ah, al
3216 00006AA2 66B80B0B          <1>       mov    ax, CMOS_REG_B * 257
3217 00006AA6 E8DE000000        <1>       call   CMOS_READ           ; READ CURRENT ALARM REGISTER
3218 00006AAB 247F              <1>       and    al, 07Fh            ; ENSURE SET BIT TURNED OFF
3219 00006AAD 0C20              <1>       or     al, 20h                  ; TURN ON ALARM ENABLE
3220 00006AAF 86E0              <1>       xchg   ah, al              ; MOVE MASK TO OUTPUT REGISTER
3221 00006AB1 E8EC000000        <1>       call   CMOS_WRITE          ; WRITE NEW ALARM MASK
3222 00006AB6 F8                <1>       clc                        ; SET CY= 0
3223                            <1> RTC_69:
3224 00006AB7 66B80000          <1>       mov    ax, 0               ; CLEAR AX REGISTER
3225 00006ABB C3                <1>       retn                       ; RETURN WITH RESULTS IN CARRY FLAC
3226                            <1>
3227                            <1> RTC_70:                          ; RESET ALARM
3228                            <1>       ;mov   al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
3229                            <1>       ;mov   ah, al
3230 00006ABC 66B80B0B          <1>       mov    ax, CMOS_REG_B * 257      ; ADDRESS ALARM REGISTER (TO BOTH AH,AL)
3231 00006AC0 E8C4000000        <1>       call   CMOS_READ           ; READ ALARM REGISTER
3232 00006AC5 2457              <1>       and    al, 57h                  ; TURN OFF ALARM ENABLE
3233 00006AC7 86E0              <1>       xchg   ah, al              ; SAVE DATA AND RECOVER ADDRESS
3234 00006AC9 E8D4000000        <1>       call   CMOS_WRITE          ; RESTORE NEW VALUE
3235 00006ACE F8                <1>       clc                        ; SET CY= 0
3236 00006ACF C3                <1>       retn                       ; RETURN WITH NO CARRY
3237                            <1>
3238                            <1> RTC_STA:                  ; INITIALIZE REAL TIME CLOCK
3239                            <1>       ;mov   al, CMOS_REG_A            ; ADDRESS REGISTER A AND LOAD DATA MASK
3240                            <1>       ;mov   ah, 26h
3241 00006AD0 66B80A26          <1>       mov    ax, (26h*100h)+CMOS_REG_A
3242 00006AD4 E8C9000000        <1>       call   CMOS_WRITE          ; INITIALIZE STATUS REGISTER A
3243                            <1>       ;mov   al, CMOS_REG_B            ; SET "SET BIT" FOR CLOCK INITIALIZATION
3244                            <1>       ;mov   ah, 82h
3245 00006AD9 66B80B82          <1>       mov    ax, (82h*100h)+CMOS_REG_B
3246 00006ADD E8C0000000        <1>       call   CMOS_WRITE          ; AND 24 HOUR MODE TO REGISTER B
3247 00006AE2 B00C              <1>       mov    al, CMOS_REG_C            ; ADDRESS REGISTER C
3248 00006AE4 E8A0000000        <1>       call   CMOS_READ           ; READ REGISTER C TO INITIALIZE
3249 00006AE9 B00D              <1>       mov    al, CMOS_REG_D            ; ADDRESS REGISTER D
3250 00006AEB E899000000        <1>       call   CMOS_READ           ; READ REGISTER D TO INITIALIZE
3251 00006AF0 C3                <1>       retn
3252                            <1>
3253                            <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
3254                            <1>
3255                            <1> ;--- HARDWARE INT  70 H -- ( IRQ LEVEL  8) ------------------------------------
3256                            <1> ; ALARM INTERRUPT HANDLER (RTC)                                          :
3257                            <1> ;       THIS ROUTINE HANDLES THE PERIODIC AND ALARM INTERRUPTS FROM THE CMOS  :
3258                            <1> ;       TIMER. INPUT FREQUENCY IS 1.024 KHZ OR APPROXIMATELY 1024 INTERRUPTS  :
3259                            <1> ;       EVERY SECOND FOR THE PERIODIC INTERRUPT. FOR THE ALARM FUNCTION,      :
3260                            <1> ;       THE INTERRUPT WILL OCCUR AT THE DESIGNATED TIME.              :
3261                            <1> ;                                                                        :
3262                            <1> ;       INTERRUPTS ARE ENABLED WHEN THE EVENT OR ALARM FUNCTION IS ACTIVATED. :
3263                            <1> ;       FOR THE EVENT INTERRUPT, THE HANDLER WILL DECREMENT THE WAIT COUNTER  :
3264                            <1> ;       AND WHEN IT EXPIRES WILL SET THE DESIGNATED LOCATION TO 80H. FOR      :
3265                            <1> ;       THE ALARM INTERRUPT. THE USER MUST PROVIDE A ROUTINE TO INTERCEPT     :
3266                            <1> ;       THE CORRECT ADDRESS FROM THE VECTOR TABLE INVOKED BY INTERRUPT 4AH    :
3267                            <1> ;       PRIOR TO SETTING THE REAL TIME CLOCK ALARM (INT 1AH, AH= 06H).        :
3268                            <1> ;--------------------------------------------------------------------------------
3269                            <1>
3270                            <1> RTC_A_INT: ; 07/01/2017
3271                            <1> ;RTC_INT:                        ; ALARM INTERRUPT
3272 00006AF1 1E                <1>       push   ds                  ; LEAVE INTERRUPTS DISABLED
3273 00006AF2 50                <1>       push   eax                 ; SAVE REGISTERS
3274 00006AF3 57                <1>       push   edi
3275                            <1> RTC_I_1:                         ; CHECK FOR SECOND INTERRUPT
3276 00006AF4 66B88C8B          <1>       mov    ax, 256*(CMOS_REG_B+NMI)+CMOS_REG_C+NMI ; ALARM AND STATUS
3277 00006AF8 E670              <1>       out    CMOS_PORT, al       ; WRITE ALARM FLAG MASK ADDRESS
3278 00006AFA 90                <1>       nop                        ; I/O DELAY
3279 00006AFB EB00              <1>       jmp    short $+2
3280 00006AFD E471              <1>       in     al, CMOS_DATA       ; READ AND RESET INTERRUPT REQUEST FLAGS
3281 00006AFF A860              <1>       test   al, 01100000b       ; CHECK FOR EITHER INTERRUPT PENDING
3282 00006B01 745D              <1>       jz     short  RTC_I_9            ; EXIT IF NOT A VALID RTC INTERRUPT
3283                            <1>
3284 00006B03 86E0              <1>       xchg   ah, al              ; SAVE FLAGS AND GET ENABLE ADDRESS
3285 00006B05 E670              <1>       out    CMOS_PORT, al       ; WRITE ALARM ENABLE MASK ADDRESS
3286 00006B07 90                <1>       nop                        ; I/O DELAY
3287 00006B08 EB00              <1>       jmp    short $+2
3288 00006B0A E471              <1>       in     al, CMOS_DATA       ; READ CURRENT ALARM ENABLE MASK
3289 00006B0C 20E0              <1>       and    al, ah              ; ALLOW ONLY SOURCES THAT ARE ENABLED
3290 00006B0E A840              <1>       test   al, 01000000b       ; CHECK FOR PERIODIC INTERRUPT
3291 00006B10 743B              <1>       jz     short RTC_I_5       ; SKIP IF NOT A PERIODIC INTERRUPT
3292                            <1>
3293                            <1> ;-----       DECREMENT WAIT COUNT BY INTERRUPT INTERVAL
3294                            <1>
3295 00006B12 66BF1000          <1>       mov    di, KDATA           ; kernel data segment
3296 00006B16 8EDF              <1>       mov    ds, di
3297                            <1>
3298 00006B18 812D[68890100]D003- <1>     sub    dword [RTC_LH], 976 ; DECREMENT COUNT BY 1/1024
3298 00006B20 0000              <1>
3299 00006B22 7329              <1>       jnc    short RTC_I_5       ; SKIP TILL 32 BIT WORD LESS THAN ZERO
3300                            <1>
3301                            <1> ;-----       TURN OFF PERIODIC INTERRUPT ENABLE
3302                            <1>
3303 00006B24 6650              <1>       push   ax                  ; SAVE INTERRUPT FLAG MASK
3304 00006B26 66B88B8B          <1>       mov    ax, 257*(CMOS_REG_B+NMI) ; INTERRUPT ENABLE REGISTER
3305 00006B2A E670              <1>       out    CMOS_PORT, al       ; WRITE ADDRESS TO CMOS CLOCK
3306 00006B2C 90                <1>       nop                        ; I/O DELAY
3307 00006B2D EB00              <1>       jmp    short $+2
3308 00006B2F E471              <1>       in     al, CMOS_DATA       ; READ CURRENT ENABLES
3309 00006B31 24BF              <1>       and    al, 0BFh            ; TURN OFF PIE
3310 00006B33 86C4              <1>       xchg   al, ah              ; GET CMOS ADDRESS AND SAVE VALUE
3311 00006B35 E670              <1>       out    CMOS_PORT, al       ; ADDRESS REGISTER B
3312 00006B37 86C4              <1>       xchg   al, ah              ; GET NEW INTERRUPT ENABLE MASK
3313 00006B39 E671              <1>       out    CMOS_DATA, al       ; SET MASK IN INTERRUPT ENABLE REGISTER
3314 00006B3B C605[6C890100]00  <1>       mov    byte [RTC_WAIT_FLAG], 0   ; SET FUNCTION ACTIVE FLAG OFF
3315 00006B42 8B3D[6D890100]    <1>       mov    edi, [USER_FLAG]    ; SET UP (DS:DI) TO POINT TO USER FLAG
3316 00006B48 C60780            <1>       mov    byte [edi], 80h           ; TURN ON USERS FLAG
3317 00006B4B 6658              <1>       pop    ax                  ; GET INTERRUPT SOURCE BACK
3318                            <1> RTC_I_5:
```

```
3319 00006B4D A820              <1>       test   al, 00100000b       ; TEST FOR ALARM INTERRUPT
3320 00006B4F 740D              <1>       jz     short RTC_I_7       ; SKIP USER INTERRUPT CALL IF NOT ALARM
3321                            <1>
3322 00006B51 B00D              <1>       mov    al, CMOS_REG_D            ; POINT TO DEFAULT READ ONLY REGISTER
3323 00006B53 E670              <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
3324 00006B55 FB                <1>       sti                        ; INTERRUPTS BACK ON NOW
3325 00006B56 52                <1>       push   edx
3326 00006B57 E81EC10000        <1>       call   INT4Ah              ; TRANSFER TO USER ROUTINE
3327 00006B5C 5A                <1>       pop    edx
3328 00006B5D FA                <1>       cli                        ; BLOCK INTERRUPT FOR RETRY
3329                            <1> RTC_I_7:                       ; RESTART ROUTINE TO HANDLE DELAYED
3330 00006B5E EB94              <1>       jmp    short RTC_I_1       ;  ENTRY AND SECOND EVENT BEFORE DONE
3331                            <1>
3332                            <1> RTC_I_9:                       ; EXIT - NO PENDING INTERRUPTS
3333 00006B60 B00D              <1>       mov    al, CMOS_REG_D            ; POINT TO DEFAULT READ ONLY REGISTER
3334 00006B62 E670              <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
3335 00006B64 B020              <1>       mov    al, EOI                   ; END OF INTERRUPT MASK TO 8259 - 2
3336 00006B66 E6A0              <1>       out    INTB00, al          ; TO 8259 - 2
3337 00006B68 E620              <1>       out    INTA00,      al           ; TO 8259 - 1
3338 00006B6A 5F                <1>       pop    edi                ; RESTORE REGISTERS
3339 00006B6B 58                <1>       pop    eax
3340 00006B6C 1F                <1>       pop    ds
3341 00006B6D CF                <1>       iretd                      ; END OF INTERRUPT
3342                            <1>
3343                            <1>
3344                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
3345                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
3346                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (bios2.asm)
3347                            <1> UPD_IPR:                       ; WAIT TILL UPDATE NOT IN PROGRESS
3348 00006B6E 51                <1>       push   ecx
3349                            <1>
3350                            <1>       ; 29/05/2016
3351 00006B6F B968110000        <1>       mov    ecx, ((1984+244)*4)/2      ; AWARD BIOS 1999, ATIME.ASM
3352                            <1>                                  ; 'WAITCPU_CK_UD_STAT'
3353                            <1>                                  ; (244Us + 1984Us)
3354                            <1>                                  ; (assume each read takes
3355                            <1>                                  ;  2 microseconds).
3356                            <1>       ;mov   ecx, 65535
3357                            <1>              ;mov cx, 800        ; SET TIMEOUT LOOP COUNT (= 800)
3358                            <1> UPD_10:
3359 00006B74 B00A              <1>       mov    al, CMOS_REG_A            ; ADDRESS STATUS REGISTER A
3360 00006B76 FA                <1>       cli                        ; NO TIMER INTERRUPTS DURING UPDATES
3361 00006B77 E80D000000        <1>       call   CMOS_READ           ; READ UPDATE IN PROCESS FLAG
3362 00006B7C A880              <1>       test   al, 80h                   ; IF UIP BIT IS ON ( CANNOT READ TIME )
3363 00006B7E 7406              <1>       jz     short UPD_90        ; EXIT WITH CY= 0 IF CAN READ CLOCK NOW
3364 00006B80 FB                <1>       sti                        ; ALLOW INTERRUPTS WHILE WAITING
3365 00006B81 E2F1              <1>       loop   UPD_10              ; LOOP TILL READY OR TIMEOUT
3366 00006B83 31C0              <1>       xor    eax, eax            ; CLEAR RESULTS IF ERROR
3367                            <1>              ; xor ax, ax
3368 00006B85 F9                <1>       stc                        ; SET CARRY FOR ERROR
3369                            <1> UPD_90:
3370 00006B86 59                <1>       pop    ecx                 ; RESTORE CALLERS REGISTER
3371 00006B87 FA                <1>       cli                        ; INTERRUPTS OFF DURING SET
3372 00006B88 C3                <1>       retn                       ; RETURN WITH CY FLAG SET
3373                            <1>
3374                            <1>
3375                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
3376                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
3377                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (test4.asm)
3378                            <1>
3379                            <1> ;--- CMOS_READ --------------------------------------------------------------
3380                            <1> ;           READ BYTE FROM CMOS_SYSTEM CLOCK CONFIGURATION TABLE       :
3381                            <1> ;                                                                      :
3382                            <1> ; INPUT: (AL)=     CMOS_TABLE ADDRESS TO BE READ                           :
3383                            <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
3384                            <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO READ               :
3385                            <1> ;                                                                      :
3386                            <1> ; OUTPUT: (AL)     VALUE AT LOCATION (AL) MOVED INTO (AL). IF BIT 7 OF (AL) WAS   :
3387                            <1> ;           ON THEN NMI LEFT DISABLED, DURING THE CMOS READ BOTH NMI AND   :
3388                            <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
3389                            <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
3390                            <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.       :
3391                            <1> ;           ONLY THE (AL) REGISTER AND THE NMI STATE IS CHANGED.       :
3392                            <1> ;--------------------------------------------------------------------------
3393                            <1>
3394                            <1> CMOS_READ:
3395 00006B89 9C                <1>       pushf                      ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
3396 00006B8A D0C0              <1>       rol    al, 1               ; MOVE NMI BIT TO LOW POSITION
3397 00006B8C F9                <1>       stc                        ; FORCE NMI BIT ON IN CARRY FLAG
3398 00006B8D D0D8              <1>       rcr    al, 1               ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
3399 00006B8F FA                <1>       cli                        ; DISABLE INTERRUPTS
3400 00006B90 E670              <1>       out    CMOS_PORT, al       ; ADDRESS LOCATION AND DISABLE NMI
3401                            <1>       ; 29/05/2016
3402                            <1>       ;nop                       ; I/O DELAY
3403 00006B92 E6EB              <1>       out    0ebh,al      ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
3404                            <1>       ;
3405 00006B94 E471              <1>       in     al, CMOS_DATA       ; READ THE REQUESTED CMOS LOCATION
3406 00006B96 6650              <1>       push   ax                  ; SAVE (AH) REGISTER VALUE AND CMOS BYTE
3407                            <1>       ; 15/03/2015 ; IBM PC/XT Model 286 BIOS source code
3408                            <1>                    ; ----- 10/06/85 (test4.asm)
3409 00006B98 B01E              <1>       mov    al, CMOS_SHUT_DOWN*2       ; GET ADDRESS OF DEFAULT LOCATION
3410                            <1>       ;mov   al, CMOS_REG_D*2    ; GET ADDRESS OF DEFAULT LOCATION
3411 00006B9A D0D8              <1>       rcr    al, 1               ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
3412 00006B9C E670              <1>       out    CMOS_PORT, al       ; SET DEFAULT TO READ ONLY REGISTER
3413 00006B9E 6658              <1>       pop    ax                  ; RESTORE (AH) AND (AL), CMOS BYTE
3414 00006BA0 9D                <1>       popf
3415 00006BA1 C3                <1>       retn                       ; RETURN WITH FLAGS RESTORED
3416                            <1>
3417                            <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
3418                            <1>
3419                            <1> ;--- CMOS_WRITE -------------------------------------------------------------
3420                            <1> ;     WRITE BYTE TO CMOS SYSTEM CLOCK CONFIGURATION TABLE              :
3421                            <1> ;                                                                      :
3422                            <1> ; INPUT: (AL)=     CMOS TABLE ADDRESS TO BE WRITTEN TO                     :
3423                            <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
```

```
3424                            <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO WRITE              :
3425                            <1> ;      (AH)= NEW VALUE TO BE PLACED IN THE ADDRESSED TABLE LOCATION           :
3426                            <1> ;                                                                         :
3427                            <1> ; OUTPUT:   VALUE IN (AH) PLACED IN LOCATION (AL) WITH NMI LEFT DISABLED   :
3428                            <1> ;           IF BIT 7 OF (AL) IS ON, DURING THE CMOS UPDATE BOTH NMI AND    :
3429                            <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
3430                            <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
3431                            <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.      :
3432                            <1> ;           ONLY THE CMOS LOCATION AND THE NMI STATE IS CHANGED.       :
3433                            <1> ;-------------------------------------------------------------------------------
3434                            <1>
3435                            <1> CMOS_WRITE:                     ; WRITE (AH) TO LOCATION (AL)
3436 00006BA2 9C               <1>         pushf                   ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
3437 00006BA3 6650             <1>         push    ax              ; SAVE WORK REGISTER VALUES
3438 00006BA5 D0C0             <1>         rol     al, 1           ; MOVE NMI BIT TO LOW POSITION
3439 00006BA7 F9               <1>         stc                     ; FORCE NMI BIT ON IN CARRY FLAG
3440 00006BA8 D0D8             <1>         rcr     al, 1           ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
3441 00006BAA FA               <1>         cli                     ; DISABLE INTERRUPTS
3442 00006BAB E670             <1>         out     CMOS_PORT, al   ; ADDRESS LOCATION AND DISABLE NMI
3443 00006BAD 88E0             <1>         mov     al, ah          ; GET THE DATA BYTE TO WRITE
3444 00006BAF E671             <1>         out     CMOS_DATA, al   ; PLACE IN REQUESTED CMOS LOCATION
3445 00006BB1 B01E             <1>         mov     al, CMOS_SHUT_DOWN*2        ; GET ADDRESS OF DEFAULT LOCATION
3446                            <1>         ;mov    al, CMOS_REG_D*2    ; GET ADDRESS OF DEFAULT LOCATION
3447 00006BB3 D0D8             <1>         rcr     al, 1           ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
3448 00006BB5 E670             <1>         out     CMOS_PORT, al   ; SET DEFAULT TO READ ONLY REGISTER
3449 00006BB7 90               <1>         nop                     ; I/O DELAY
3450 00006BB8 E471             <1>         in      al, CMOS_DATA   ; OPEN STANDBY LATCH
3451 00006BBA 6658             <1>         pop     ax              ; RESTORE WORK REGISTERS
3452 00006BBC 9D               <1>         popf
3453 00006BBD C3               <1>         retn
3454                            <1>
3455                            <1> ; /// End Of TIMER FUNCTIONS ///
2894
2895 00006BBE 90<rep 2h>                   Align 16
2896
2897                                        gdt:  ; Global Descriptor Table
2898                                              ; 02/12/2020
2899                                              ; (30/07/2015, conforming cs)
2900                                              ; (26/03/2015)
2901                                              ; (24/03/2015, tss)
2902                                              ; (19/03/2015)
2903                                              ; (29/12/2013)
2904                                              ;
2905 00006BC0 0000000000000000                   dw 0, 0, 0, 0       ; NULL descriptor
2906                                        gdt_kcode:
2907                                              ; 18/08/2014
2908                                                                  ; 8h kernel code segment, base = 00000000h
2909                                              ;dw 0FFFFh, 0, 9E00h, 00CFh      ; KCODE  ; 30/12/2016
2910 00006BC8 FFFF0000009ACF00                   dw 0FFFFh, 0, 9A00h, 00CFh; KCODE
2911                                        gdt_kdata:
2912                                                                  ; 10h kernel data segment, base = 00000000h
2913 00006BD0 FFFF00000092CF00                   dw 0FFFFh, 0, 9200h, 00CFh; KDATA
2914                                        gdt_ucode:
2915                                                                  ; 1Bh user code segment, base address = 400000h ; CORE
2916                                              ;dw 0FBFFh, 0, 0FE40h, 00CFh     ; UCODE  ; 30/12/2016
2917 00006BD8 FFFB000040FACF00                   dw 0FBFFh, 0, 0FA40h, 00CFh      ; UCODE
2918                                        gdt_udata:
2919                                                                  ; 23h user data segment, base address = 400000h ; CORE
2920 00006BE0 FFFB000040F2CF00                   dw 0FBFFh, 0, 0F240h, 00CFh      ; UDATA
2921                                        gdt_tss:
2922                                                                  ; Task State Segment
2923 00006BE8 6700                               dw 0067h ; Limit = 103 ; (104-1, tss size = 104 byte,
2924                                                                   ;  no IO permission in ring 3)
2925                                        gdt_tss0:
2926 00006BEA 0000                               dw 0  ; TSS base address, bits 0-15
2927                                        gdt_tss1:
2928 00006BEC 00                                 db 0  ; TSS base address, bits 16-23
2929                                                                  ; 49h
2930 00006BED E9                                 db 11101001b ; 0E9h => P=1/DPL=11/0/1/0/B/1 --> B = Task is busy (1)
2931 00006BEE 00                                 db 0 ; G/0/0/AVL/LIMIT=0000 ; (Limit bits 16-19 = 0000) (G=0, 1 byte)
2932                                        gdt_tss2:
2933 00006BEF 00                                 db 0  ; TSS base address, bits 24-31
2934
2935                                                  ; 30/11/2020
2936                                                  ; 29/11/2020 - TRDOS v2.0.3
2937                                                  ; VESA VBE3 VIDE BIOS 32 BIT PMI SEGMENTS (16 bit segments)
2938                                                                  ; 30h ; VBE3CS
2939                                        _vbe3_CS:  ; vesa vbe3 bios uses this as code seg (same addr with _vbe3_DS)
2940                                                  ; limit = 65536, base addr = 0, P/DPL/1/Type/C/R/A = 9Ah, 16 bit
2941 00006BF0 FFFF0000009A0000                       dw 0FFFFh, 0, 9A00h, 0 ; Note: base addr will be initialized
2942                                                                  ; 38h ; VBE3BDS
2943                                        _vbe3_BDS: ; vesa vbe3 bios uses this as equivalent of rombios data segment
2944                                                  ; limit = 1536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
2945 00006BF8 FF05000000920000                       dw 05FFh, 0, 9200h, 0 ; Note: base addr will be initialized
2946                                                                  ; 40h ; VBE3A000
2947                                        _A0000Sel: ; VGA default video memory address
2948                                                  ; limit = 65536, base addr = 0A0000h, 16 bit
2949 00006C00 FFFF00000A920000                       dw 0FFFFh, 0, 920Ah, 0
2950                                                                  ; 48h ; VBE3B000
2951                                        _B0000Sel: ; MDA (monochrome) video memory address
2952                                                  ; limit = 65536, base addr = 0B0000h, 16 bit
2953 00006C08 FFFF00000B920000                       dw 0FFFFh, 0, 920Bh, 0
2954                                                                  ; 50h ; VBE3B800
2955                                        _B8000Sel: ; CGA video memory address
2956                                                  ; limit = 32768, base addr = 0B8000h, 16 bit
2957 00006C10 FF7F00800B920000                       dw 07FFFh, 8000h, 920Bh, 0
2958                                                                  ; 58h ; VBE3DS
2959                                        _vbe3_DS: ; vesa vbe3 bios uses this as data seg (CodeSegSel in PMInfoBlock)
2960                                                  ; limit = 65536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
2961 00006C18 FFFF000000920000                       dw 0FFFFh, 0, 9200h, 0 ; Note: base addr will be initialized
2962                                                                  ; 60h ; VBE3SS
2963                                        _vbe3_SS: ; kernel's stack segment but 16 bit version (same stack addr)
2964                                                  ; limit = 1024, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
2965 00006C20 FF03000000920000                       dw 03FFh, 0, 9200h, 0 ; Note: base addr will be initialized
2966                                                                  ; 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
00006C28 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
00006C30 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:
00006C38 7700                         dw gdt_end - gdt - 1    ; Limit (size)
00006C3A [C06B0000]                   dd gdt                  ; Address of the GDT

                                      ; 20/08/2014
                                idtd:
00006C3E 7F02                         dw idt_end - idt - 1    ; Limit (size)
00006C40 [08860100]                   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>
00006C44 [A76C0000]           <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:
00006C48 01             <1>             DB      01              ; DRIVE TYPE, MEDIA TABLE
                        <1>                ;DW      MD_TBL1
00006C49 [666C0000]     <1>             dd      MD_TBL1
00006C4D 82             <1>             DB      02+BIT7ON
                        <1>             ;DW     MD_TBL2
00006C4E [736C0000]     <1>                dd      MD_TBL2
00006C52 02             <1> DR_DEFAULT:  DB     02
                        <1>                ;DW      MD_TBL3
00006C53 [806C0000]     <1>             dd      MD_TBL3
00006C57 03             <1>             DB      03
                        <1>                ;DW      MD_TBL4
00006C58 [8D6C0000]     <1>             dd      MD_TBL4
00006C5C 84             <1>             DB      04+BIT7ON
                        <1>                ;DW      MD_TBL5
00006C5D [9A6C0000]     <1>             dd      MD_TBL5
00006C61 04             <1>             DB      04
                        <1>                ;DW      MD_TBL6
00006C62 [A76C0000]     <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:
00006C66 DF             <1>         DB      11011111B       ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006C67 02             <1>         DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006C68 25             <1>         DB      MOTOR_WAIT      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006C69 02             <1>         DB      2               ; 512 BYTES/SECTOR
00006C6A 09             <1>         DB      09              ; EOT (LAST SECTOR ON TRACK)
00006C6B 2A             <1>         DB      02AH            ; GAP LENGTH
00006C6C FF             <1>         DB      0FFH            ; DTL
00006C6D 50             <1>         DB      050H            ; GAP LENGTH FOR FORMAT
00006C6E F6             <1>         DB      0F6H            ; FILL BYTE FOR FORMAT
00006C6F 0F             <1>         DB      15              ; HEAD SETTLE TIME (MILLISECONDS)
```

```
3177 00006C70 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
3178 00006C71 27                <1>       DB     39           ; MAX. TRACK NUMBER
3179 00006C72 80                <1>       DB     RATE_250     ; DATA TRANSFER RATE
3180                            <1> ;--------------------------------------------------------
3181                            <1> ;     360 KB MEDIA IN 1.2 MB DRIVE                  :
3182                            <1> ;--------------------------------------------------------
3183                            <1> MD_TBL2:
3184 00006C73 DF                <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
3185 00006C74 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
3186 00006C75 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
3187 00006C76 02                <1>       DB     2            ; 512 BYTES/SECTOR
3188 00006C77 09                <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
3189 00006C78 2A                <1>       DB     02AH         ; GAP LENGTH
3190 00006C79 FF                <1>       DB     0FFH         ; DTL
3191 00006C7A 50                <1>       DB     050H         ; GAP LENGTH FOR FORMAT
3192 00006C7B F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
3193 00006C7C 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
3194 00006C7D 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
3195 00006C7E 27                <1>       DB     39           ; MAX. TRACK NUMBER
3196 00006C7F 40                <1>       DB     RATE_300     ; DATA TRANSFER RATE
3197                            <1> ;--------------------------------------------------------
3198                            <1> ;     1.2 MB MEDIA IN 1.2 MB DRIVE                  :
3199                            <1> ;--------------------------------------------------------
3200                            <1> MD_TBL3:
3201 00006C80 DF                <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
3202 00006C81 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
3203 00006C82 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
3204 00006C83 02                <1>       DB     2            ; 512 BYTES/SECTOR
3205 00006C84 0F                <1>       DB     15           ; EOT (LAST SECTOR ON TRACK)
3206 00006C85 1B                <1>       DB     01BH         ; GAP LENGTH
3207 00006C86 FF                <1>       DB     0FFH         ; DTL
3208 00006C87 54                <1>       DB     054H         ; GAP LENGTH FOR FORMAT
3209 00006C88 F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
3210 00006C89 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
3211 00006C8A 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
3212 00006C8B 4F                <1>       DB     79           ; MAX. TRACK NUMBER
3213 00006C8C 00                <1>       DB     RATE_500     ; DATA TRANSFER RATE
3214                            <1> ;--------------------------------------------------------
3215                            <1> ;     720 KB MEDIA IN 720 KB DRIVE                  :
3216                            <1> ;--------------------------------------------------------
3217                            <1> MD_TBL4:
3218 00006C8D DF                <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
3219 00006C8E 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
3220 00006C8F 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
3221 00006C90 02                <1>       DB     2            ; 512 BYTES/SECTOR
3222 00006C91 09                <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
3223 00006C92 2A                <1>       DB     02AH         ; GAP LENGTH
3224 00006C93 FF                <1>       DB     0FFH         ; DTL
3225 00006C94 50                <1>       DB     050H         ; GAP LENGTH FOR FORMAT
3226 00006C95 F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
3227 00006C96 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
3228 00006C97 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
3229 00006C98 4F                <1>       DB     79           ; MAX. TRACK NUMBER
3230 00006C99 80                <1>       DB     RATE_250     ; DATA TRANSFER RATE
3231                            <1> ;--------------------------------------------------------
3232                            <1> ;     720 KB MEDIA IN 1.44 MB DRIVE                 :
3233                            <1> ;--------------------------------------------------------
3234                            <1> MD_TBL5:
3235 00006C9A DF                <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
3236 00006C9B 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
3237 00006C9C 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
3238 00006C9D 02                <1>       DB     2            ; 512 BYTES/SECTOR
3239 00006C9E 09                <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
3240 00006C9F 2A                <1>       DB     02AH         ; GAP LENGTH
3241 00006CA0 FF                <1>       DB     0FFH         ; DTL
3242 00006CA1 50                <1>       DB     050H         ; GAP LENGTH FOR FORMAT
3243 00006CA2 F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
3244 00006CA3 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
3245 00006CA4 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
3246 00006CA5 4F                <1>       DB     79           ; MAX. TRACK NUMBER
3247 00006CA6 80                <1>       DB     RATE_250     ; DATA TRANSFER RATE
3248                            <1> ;--------------------------------------------------------
3249                            <1> ;     1.44 MB MEDIA IN 1.44 MB DRIVE                :
3250                            <1> ;--------------------------------------------------------
3251                            <1> MD_TBL6:
3252 00006CA7 AF                <1>       DB     10101111B    ; SRT=A, HD UNLOAD=0F - 1ST SPECIFY BYTE
3253 00006CA8 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
3254 00006CA9 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
3255 00006CAA 02                <1>       DB     2            ; 512 BYTES/SECTOR
3256 00006CAB 12                <1>       DB     18           ; EOT (LAST SECTOR ON TRACK)
3257 00006CAC 1B                <1>       DB     01BH         ; GAP LENGTH
3258 00006CAD FF                <1>       DB     0FFH         ; DTL
3259 00006CAE 6C                <1>       DB     06CH         ; GAP LENGTH FOR FORMAT
3260 00006CAF F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
3261 00006CB0 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
3262 00006CB1 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
3263 00006CB2 4F                <1>       DB     79           ; MAX. TRACK NUMBER
3264 00006CB3 00                <1>       DB     RATE_500     ; DATA TRANSFER RATE
3265                            <1>
3266                            <1>
3267                            <1> ; << diskette.inc >>
3268                            <1> ; +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
3269                            <1> ;
3270                            <1> ;----------------------------------------
3271                            <1> ;     ROM BIOS DATA AREAS       :
3272                            <1> ;----------------------------------------
3273                            <1>
3274                            <1> ;DATA       SEGMENT AT 40H            ; ADDRESS= 0040:0000
3275                            <1>
3276                            <1> ;----------------------------------------
3277                            <1> ;     FIXED DISK DATA AREAS           :
3278                            <1> ;----------------------------------------
3279                            <1>
3280                            <1> ;DISK_STATUS1:     DB     0           ; FIXED DISK STATUS
3281                            <1> ;HF_NUM:           DB     0     ; COUNT OF FIXED DISK DRIVES
```

```
                        <1> ;CONTROL_BYTE:     DB     0            ; HEAD CONTROL BYTE
                        <1> ;@PORT_OFF   DB    ?            ; RESERVED (PORT OFFSET)
                        <1>
                        <1> ;----------------------------------------
                        <1> ;     ADDITIONAL MEDIA DATA            :
                        <1> ;----------------------------------------
                        <1>
                        <1> ;@LASTRATE   DB    ?            ; LAST DISKETTE DATA RATE SELECTED
                        <1> ;HF_STATUS   DB    0            ; STATUS REGISTER
                        <1> ;HF_ERROR    DB    0            ; ERROR REGISTER
                        <1> ;HF_INT_FLAG DB    0            ; FIXED DISK INTERRUPT FLAG
                        <1> ;HF_CNTRL    DB    0            ; COMBO FIXED DISK/DISKETTE CARD BIT 0=1
                        <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>
                        <1> ERR_TBL:
00006CB4 E0             <1>       db     NO_ERR
00006CB5 024001BB       <1>       db     BAD_ADDR_MARK,BAD_SEEK,BAD_CMD,UNDEF_ERR
00006CB9 04BB100A       <1>       db     RECORD_NOT_FND,UNDEF_ERR,BAD_ECC,BAD_SECTOR
                        <1>
                        <1> ; 17/12/2014 (mov ax, [cfd])
                        <1> ; 11/12/2014
00006CBD 00             <1> cfd:         db     0            ; current floppy drive (for GET_PARM)
                        <1> ; 17/12/2014                     ; instead of 'DISK_POINTER'
00006CBE 01             <1> pfd:         db     1            ; previous floppy drive (for GET_PARM)
                        <1>                                  ; (initial value of 'pfd
                        <1>                                  ; must be different then 'cfd' value
                        <1>                                  ; to force updating/initializing
                        <1>                                  ; current drive parameters)
00006CBF 90             <1> align 2
                        <1>
00006CC0 F001           <1> HF_PORT:     dw     1F0h  ; Default = 1F0h
                        <1>                            ; (170h)
00006CC2 F603           <1> HF_REG_PORT: dw     3F6h  ; HF_PORT + 206h
                        <1>
                        <1> ; 05/01/2015
00006CC4 00             <1> hf_m_s:         db      0     ; (0 = Master, 1 = Slave)
                        <1>
                        <1> ; ****************************************************************************

00006CC5 90                 Align 2

                            ; 04/11/2014 (Retro UNIX 386 v1)
00006CC6 0000               mem_1m_1k:   dw 0  ; Number of contiguous KB between
                                               ; 1 and 16 MB, max. 3C00h = 15 MB.
00006CC8 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)
00006CCA 00                 boot_drv:    db 0 ; boot drive number (physical)
                            ; 24/11/2014
00006CCB 00                 drv:       db 0
00006CCC 00                 last_drv:    db 0 ; last hdd
00006CCD 00                 hdc:         db 0  ; number of hard disk drives
                                             ; (present/detected)

                            ; 24/11/2014 (Retro UNIX 386 v1)
                            ; Physical drive type & flags
00006CCE 00                 fd0_type:    db 0  ; floppy drive type
00006CCF 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)
00006CD0 00                 hd0_type:    db 0  ; EDD status for hd0 (bit 7 = present flag)
00006CD1 00                 hd1_type:    db 0  ; EDD status for hd1 (bit 7 = present flag)
00006CD2 00                 hd2_type:    db 0  ; EDD status for hd2 (bit 7 = present flag)
00006CD3 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
00006CD4 00000000000000000-     drv.cylinders: dw 0,0,0,0,0,0,0
00006CDD 0000000000
00006CE2 00000000000000000-     drv.heads:     dw 0,0,0,0,0,0,0
00006CEB 0000000000
00006CF0 00000000000000000-     drv.spt:       dw 0,0,0,0,0,0,0
00006CF9 0000000000
00006CFE 00000000000000000-     drv.size:      dd 0,0,0,0,0,0,0
00006D07 00000000000000000-
00006D10 00000000000000000-
00006D19 00
00006D1A 00000000000000         drv.status:    db 0,0,0,0,0,0,0
00006D21 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
00006D28 524F50514B             <1> K30:  db    82,79,80,81,75
00006D2D 4C4D474849             <1>       db    76,77,71,72,73            ; 10 NUMBER ON KEYPAD
                                <1> ;-----      SUPER-SHIFT-TABLE
00006D32 101112131415           <1>       db    16,17,18,19,20,21   ; A-Z TYPEWRITER CHARS
00006D38 161718191E1F           <1>       db    22,23,24,25,30,31
00006D3E 202122232425           <1>       db    32,33,34,35,36,37
00006D44 262C2D2E2F30           <1>       db    38,44,45,46,47,48
00006D4A 3132                   <1>       db    49,50
                                <1>
                                <1> ;-----      TABLE OF SHIFT KEYS AND MASK VALUES
                                <1> ;-----      KEY_TABLE
00006D4C 52                     <1> _K6:    db      INS_KEY                 ; INSERT KEY
00006D4D 3A4546381D             <1>         db    CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
00006D52 2A36                   <1>           db    LEFT_KEY,RIGHT_KEY
                                <1> _K6L    equ     $-_K6
                                <1>
                                <1> ;-----      MASK_TABLE
00006D54 80                     <1> _K7:    db      INS_SHIFT               ; INSERT MODE SHIFT
00006D55 4020100804             <1>         db    CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
00006D5A 0201                   <1>         db    LEFT_SHIFT,RIGHT_SHIFT
                                <1>
                                <1> ;-----      TABLES FOR CTRL CASE            ;---- CHARACTERS ------
00006D5C 1BFF00FFFFFF           <1> _K8:  db    27,-1,0,-1,-1,-1    ; Esc, 1, 2, 3, 4, 5
00006D62 1EFFFFFFFF1F           <1>       db    30,-1,-1,-1,-1,31   ; 6, 7, 8, 9, 0, -
00006D68 FF7FFF111705           <1>       db    -1,127,-1,17,23,5   ; =, Bksp, Tab, Q, W, E
00006D6E 12141915090F           <1>       db    18,20,25,21,9,15    ; R, T, Y, U, I, O
00006D74 101B1D0AFF01           <1>       db    16,27,29,10,-1,1    ; P, [, ], Enter, Ctrl, A
00006D7A 13040607080A           <1>       db    19,4,6,7,8,10       ; S, D, F, G, H, J
00006D80 0B0CFFFFFFFF           <1>       db    11,12,-1,-1,-1,-1   ; K, L, :, ', `, LShift
00006D86 1C1A18031602           <1>       db    28,26,24,3,22,2           ; Bkslash, Z, X, C, V, B
00006D8C 0E0DFFFFFFFF           <1>       db    14,13,-1,-1,-1,-1   ; N, M, ,, ., /, RShift
00006D92 96FF20FF               <1>       db    150,-1,' ',-1       ; *, ALT, Spc, CL
                                <1>       ;                         ;----- FUNCTIONS ------
00006D96 5E5F60616263           <1>       db    94,95,96,97,98,99   ; F1 - F6
00006D9C 64656667FFFF           <1>       db    100,101,102,103,-1,-1     ; F7 - F10, NL, SL
00006DA2 778D848E738F           <1>       db    119,141,132,142,115,143   ; Home, Up, PgUp, -, Left, Pad5
00006DA8 749075917692           <1>       db    116,144,117,145,118,146 ; Right, +, End, Down, PgDn, Ins
00006DAE 93FFFFFF898A           <1>       db    147,-1,-1,-1,137,138      ; Del, SysReq, Undef, WT, F11, F12
                                <1>
                                <1> ;-----      TABLES FOR LOWER CASE ----------
00006DB4 1B3132333435363738-    <1> K10:  db    27,'1234567890-=',8,9
00006DBD 39302D3D0809           <1>
00006DC3 71776572747975696F-    <1>       db    'qwertyuiop[]',13,-1,'asdfghjkl;',39
00006DCC 705B5D0DFF61736466-    <1>
00006DD5 67686A6B6C3B27         <1>
00006DDC 60FF5C7A786376626E-    <1>       db    96,-1,92,'zxcvbnm,./',-1,'*',-1,' ',-1
00006DE5 6D2C2E2FFF2AFF20FF     <1>
                                <1> ;-----      LC TABLE SCAN
00006DEE 3B3C3D3E3F             <1>       db    59,60,61,62,63            ; BASE STATE OF F1 - F10
00006DF3 4041424344             <1>       db    64,65,66,67,68
00006DF8 FFFF                   <1>       db    -1,-1               ; NL, SL
                                <1>
                                <1> ;-----      KEYPAD TABLE
00006DFA 474849FF4BFF           <1> K15:  db    71,72,73,-1,75,-1   ; BASE STATE OF KEYPAD KEYS
00006E00 4DFF4F50515253         <1>       db    77,-1,79,80,81,82,83
00006E07 FFFF5C8586             <1>       db    -1,-1,92,133,134    ; SysRq, Undef, WT, F11, F12
                                <1>
                                <1> ;-----      TABLES FOR UPPER CASE ----------
00006E0C 1B21402324255E262A-    <1> K11:  db    27,'!@#$%',94,'&*()_+',8,0
00006E15 28295F2B0800           <1>
00006E1B 51574552545955494F-    <1>       db    'QWERTYUIOP{}',13,-1,'ASDFGHJKL:"'
00006E24 507B7D0DFF41534446-    <1>
00006E2D 47484A4B4C3A22         <1>
00006E34 7EFF7C5A584356424E-    <1>       db    126,-1,'|ZXCVBNM<>?',-1,'*',-1,' ',-1
00006E3D 4D3C3E3FFF2AFF20FF     <1>
                                <1> ;-----      UC TABLE SCAN
00006E46 5455565758             <1> K12:  db    84,85,86,87,88            ; SHIFTED STATE OF F1 - F10
00006E4B 595A5B5C5D             <1>       db    89,90,91,92,93
00006E50 FFFF                   <1>       db    -1,-1               ; NL, SL
                                <1>
                                <1> ;-----      NUM STATE TABLE
00006E52 3738392D3435362B31-    <1> K14:  db    '789-456+1230.'           ; NUMLOCK STATE OF KEYPAD KEYS
00006E5B 3233302E               <1>
                                <1>       ;
00006E5F 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> ;----------------------------------------
00006E64 00                     <1> BIOS_BREAK   db     0            ; BIT 7=1 IF BREAK KEY HAS BEEN PRESSED
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     KEYBOARD DATA AREAS       ;
                                <1> ;----------------------------------------
                                <1>
00006E65 00                     <1> KB_FLAG            db     0            ; KEYBOARD SHIFT STATE AND STATUS FLAGS
00006E66 00                     <1> KB_FLAG_1    db     0            ; SECOND BYTE OF KEYBOARD STATUS
00006E67 00                     <1> KB_FLAG_2    db     0            ; KEYBOARD LED FLAGS
00006E68 00                     <1> KB_FLAG_3    db     0            ; KEYBOARD MODE STATE AND TYPE FLAGS
00006E69 00                     <1> ALT_INPUT    db     0            ; STORAGE FOR ALTERNATE KEY PAD ENTRY
00006E6A [7A6E0000]             <1> BUFFER_START dd     KB_BUFFER    ; OFFSET OF KEYBOARD BUFFER START
00006E6E [9A6E0000]             <1> BUFFER_END   dd     KB_BUFFER + 32      ; OFFSET OF END OF BUFFER
00006E72 [7A6E0000]             <1> BUFFER_HEAD  dd     KB_BUFFER    ; POINTER TO HEAD OF KEYBOARD BUFFER
00006E76 [7A6E0000]             <1> BUFFER_TAIL  dd     KB_BUFFER    ; POINTER TO TAIL OF KEYBOARD BUFFER
                                <1> ; ------     HEAD = TAIL  INDICATES THAT THE BUFFER IS EMPTY
00006E7A 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> ;----------------------------------------
00006E9A 03                     <1> CRT_MODE:    db     3      ; CURRENT DISPLAY MODE (TYPE)
00006E9B 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
00006E9C 50                     <1> CRT_COLS:    db     80     ; Number of columns
                                <1>
                                <1> ; 01/07/2016
00006E9D 00                     <1> CRT_PALETTE: db     0      ; Current palette setting
                                <1>
                                <1> ; 03/07/2016
00006E9E 10                     <1> CHAR_HEIGHT: db     16     ; Default character height
00006E9F 60                     <1> VGA_VIDEO_CTL:     db     60h    ; ROM BIOS DATA AREA Offset 87h
00006EA0 F9                     <1> VGA_SWITCHES:      db     0F9h   ; Feature Bit Switches (the basic screen)
00006EA1 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
00006EA2 19                     <1> VGA_ROWS:    db     25
                                <1>
                                <1> ; 16/01/2016
                                <1> chr_attrib:  ; Character color/attributes for video pages (0 to 7)
```

```
3152 00006EA3 0707070707070707    <1>       db     07h, 07h, 07h, 07h, 07h, 07h, 07h, 07h
3153                              <1> ; 30/01/2016
3154                              <1> vmode:
3155 00006EAB 0303030303030303    <1>       db     3,3,3,3,3,3,3,3 ; video modes for pseudo screens
3156                              <1>
3157                              <1> CURSOR_MODE: ; cursor start (ch) = 14, cursor end (cl) = 15
3158 00006EB3 0F0E                <1>       db     15, 14 ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
3159                              <1>
3160                              <1> ;align 4
3161                              <1> ;VGA_BASE: ; 26/07/2016
3162                              <1> ;     dd      0B8000h  ; (Mode < 0Dh) or 0A0000h (mode >= 0Dh)
3163                              <1>
3164 00006EB5 90                 <1> align 2
3165                              <1>
3166                              <1> vga_modes:
3167                              <1>       ; 25/07/2016
3168                              <1>       ; 09/07/2016
3169                              <1>       ; 03/07/2016
3170                              <1>       ; valid (implemented) video modes (>7, extension to IBM PC CGA modes)
3171 00006EB6 0302010007040506    <1>       db     03h, 02h, 01h, 00h, 07h, 04h, 05h, 06h
3172                              <1> vga_g_modes: ; 31/07/2016
3173 00006EBE 13F0126A0D0E1011    <1>       db     13h, 0F0h, 12h, 6Ah, 0Dh, 0Eh, 10h, 11h
3174                              <1> vga_mode_count equ $ - vga_modes
3175                              <1> vga_g_mode_count equ $ - vga_g_modes
3176                              <1>
3177                              <1> vga_mode_tbl_ptr:
3178                              <1>       ; 25/07/2016
3179 00006EC6 [266F0000]          <1>       dd     vga_mode_03h
3180 00006ECA [266F0000]          <1>       dd     vga_mode_03h ; mode 02h -> mode 03h
3181 00006ECE [666F0000]          <1>       dd     vga_mode_01h
3182 00006ED2 [666F0000]          <1>       dd     vga_mode_01h ; mode 00h -> mode 01h
3183                              <1>       ;dd    vga_mode_07h
3184 00006ED6 [266F0000]          <1>       dd     vga_mode_03h ; mode 07h -> mode 03h
3185 00006EDA [A66F0000]          <1>       dd     vga_mode_04h
3186 00006EDE [A66F0000]          <1>       dd     vga_mode_04h ; mode 05h -> mode 04h
3187 00006EE2 [E66F0000]          <1>       dd     vga_mode_06h
3188 00006EE6 [26700000]          <1>       dd     vga_mode_13h
3189 00006EEA [66700000]          <1>       dd     vga_mode_F0h
3190 00006EEE [A6700000]          <1>       dd     vga_mode_12h
3191 00006EF2 [E6700000]          <1>       dd     vga_mode_6Ah
3192 00006EF6 [26710000]          <1>       dd     vga_mode_0Dh
3193 00006EFA [66710000]          <1>       dd     vga_mode_0Eh
3194 00006EFE [A6710000]          <1>       dd     vga_mode_10h
3195 00006F02 [E6710000]          <1>       dd     vga_mode_11h
3196                              <1>
3197                              <1> vga_memmodel:
3198                              <1>       ; 25/07/2016
3199                              <1>       ; 07/07/2016
3200                              <1>       CTEXT  equ 0
3201                              <1>       ;MTEXT equ 1
3202                              <1>       MTEXT  equ 0 ; mode 07h -> mode 03h
3203                              <1>       CGA    equ 2
3204                              <1>       LINEAR8 equ 5
3205                              <1>       PLANAR4      equ 4
3206                              <1>       PLANAR1      equ 3
3207 00006F06 0000000000020202    <1>       db     CTEXT, CTEXT, CTEXT, CTEXT, MTEXT, CGA, CGA, CGA
3208                              <1> vga_g_memmodel: ; 31/07/2016
3209 00006F0E 0504040404040403    <1>       db     LINEAR8, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR1
3210                              <1> ;vga_pixbits:
3211                              <1> ;     ; 25/07/2016
3212                              <1> ;     ; 08/07/2016
3213                              <1> ;     db     4, 4, 4, 4, 4, 2, 2, 1, 8, 4, 4, 4, 4, 4, 4, 1
3214                              <1> vga_dac_s:
3215 00006F16 020202020001010103- <1>       db     2, 2, 2, 2, 0, 1, 1, 1, 3, 3, 2, 2, 1, 1, 2, 2
3215 00006F1F 03020201010202      <1>
3216                              <1>                                         ; (vgatables.h, VGAMODES, dac)
3217                              <1>                                         ; 17/11/2020
3218                              <1> vga_params:
3219                              <1>       ; 23/11/2020
3220                              <1>       ; 16/11/2020
3221                              <1>       ; 09/11/2020, 10/11/2020, 11/11/2020 (TRDOS 386 v2.0.3)
3222                              <1>       ; 25/07/2016
3223                              <1>       ; 19/07/2016
3224                              <1>       ; 03/07/2016
3225                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
3226                              <1>       ; vgabios-0.7a (2011)
3227                              <1>       ; by the LGPL VGABios Developers Team (2001-2008)
3228                              <1>       ; 'vgatables.h'
3229                              <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
3230                              <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
3231                              <1>
3232                              <1>       ; 09/11/2020
3233                              <1>       ; Block Structure of Video Parameter Table
3234                              <1>       ;
3235                              <1>       ; Offset     # Bytes            Contents
3236                              <1>       ;
3237                              <1>       ;  0         1            # columns
3238                              <1>       ;  1         1            # rows - 1
3239                              <1>       ;  2         1            Pixels/character
3240                              <1>       ;  3-4       2            Page length
3241                              <1>       ;  5-8       4            Sequencer Registers
3242                              <1>       ;  9         1            Miscellaneous Register
3243                              <1>       ;  10-34     25           CRTC Registers
3244                              <1>       ;  35-54     20           Attribute Registers
3245                              <1>       ;  55-63     9            Graphics Controller Registers
3246                              <1>       ;
3247                              <1>       ; Ref: Programmer's Guide to EGA, VGA, and Super VGA cards
3248                              <1>       ; (Richard F. Ferraro, 1994)
3249                              <1>
3250                              <1>       ;
3251                              <1> vga_mode_03h:  ; mode 03h, 80*25 text, CGA colors
3252                              <1>       ; 11/11/2020
3253 00006F26 5018100010          <1>       db     80, 24, 16, 00h, 10h ; tw, th-1, ch, slength (5)
3254 00006F2B 00030002            <1>       db     00h, 03h, 00h, 02h ; sequ regs (4)
3255 00006F2F 67                  <1>       db     67h    ; misc reg (1)
```

```
3256 00006F30 5F4F50825581BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
3257                              <1>       ; 09/11/2020
3258                              <1>       ;db   5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
3259 00006F38 004F                <1>       db    00h, 4Fh
3260                              <1> vga_p_cm_pos equ $ - vga_mode_03h
3261 00006F3A 0D0E00000000        <1>       db    0Dh, 0Eh, 00h, 00h, 00h, 00h
3262 00006F40 9C8E8F281F96B9A3    <1>       db    9Ch, 8Eh, 8Fh, 28h, 1Fh, 96h, 0B9h, 0A3h
3263 00006F48 FF                  <1>       db    0FFh  ; crtc_regs (25)
3264 00006F49 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
3265 00006F51 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
3266                              <1>       ;db   0Ch, 00h, 0Fh, 08h  ; actl regs (20)
3267 00006F59 0C000F00            <1>       db    0Ch, 00h, 0Fh, 00h  ; 19/11/2020
3268 00006F5D 0000000000100E0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs (9)
3269                              <1>       ; 09/11/2020
3270                              <1>       ;db   00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 00h, 0FFh ; grdc regs (9)
3271                              <1> vga_mode_01h:      ; mode 01h, 40*25 text, CGA colors
3272 00006F66 2818100008          <1>       db    40, 24, 16, 00h, 08h ; tw, th-1, ch, slength
3273 00006F6B 08030002            <1>       db    08h, 03h, 00h, 02h  ; sequ regs
3274 00006F6F 67                  <1>       db    67h   ; misc reg
3275 00006F70 2D2728902BA0BF1F    <1>       db    2Dh, 27h, 28h, 90h, 2Bh, 0A0h, 0BFh, 1Fh
3276 00006F78 004F0D0E00000000    <1>       db    00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
3277 00006F80 9C8E8F141F96B9A3    <1>       db    9Ch, 8Eh, 8Fh, 14h, 1Fh, 96h, 0B9h, 0A3h
3278 00006F88 FF                  <1>       db    0FFh  ; crtc_regs
3279 00006F89 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
3280 00006F91 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
3281                              <1>       ;db   0Ch, 00h, 0Fh, 08h  ; actl regs (20)
3282 00006F99 0C000F00            <1>       db    0Ch, 00h, 0Fh, 00h  ; 19/11/2020
3283 00006F9D 0000000000100E0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs
3284                              <1> ;vga_mode_07h:     ; mode 07h, 80*25 text, mono color
3285                              <1> ;     db    80, 24, 16, 00h, 10h  ; tw, th-1, ch, slength
3286                              <1> ;     db    00h, 03h, 00h, 02h ; sequ regs
3287                              <1> ;     db    66h   ; misc reg
3288                              <1> ;     db    5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
3289                              <1> ;     db    00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
3290                              <1> ;     db    9Ch, 8Eh, 8Fh, 28h, 0Fh, 96h, 0B9h, 0A3h
3291                              <1> ;     db    0FFh  ; crtc regs
3292                              <1> ;     db    00h, 08h, 08h, 08h, 08h, 08h, 08h, 08h
3293                              <1> ;     db    10h, 18h, 18h, 18h, 18h, 18h, 18h, 18h
3294                              <1> ;     db    0Eh, 00h, 0Fh, 08h  ; actl regs
3295                              <1> ;     db    00h, 00h, 00h, 00h, 00h, 10h, 0Ah, 0Fh, 0FFh ; grdc regs
3296                              <1> vga_mode_04h:      ; 320*200 graphics, 4 colors, CGA
3297                              <1>       ; 11/11/2020
3298 00006FA6 2818080040          <1>       db    40, 24, 8, 00h, 40h ; tw, th-1, ch, slength
3299 00006FAB 09030002            <1>       db    09h, 03h, 00h, 02h ; sequ regs
3300 00006FAF 63                  <1>       db    63h   ; misc reg
3301 00006FB0 2D2728902B80BF1F    <1>       db    2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
3302 00006FB8 00C1000000000000    <1>       db    00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
3303 00006FC0 9C8E8F140096B9A2    <1>       db    9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0A2h
3304 00006FC8 FF                  <1>       db    0FFh  ; crtc_regs
3305 00006FC9 0013151702040607    <1>       db    00h, 13h, 15h, 17h, 02h, 04h, 06h, 07h
3306 00006FD1 1011121314151617    <1>       db    10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
3307 00006FD9 01000300            <1>       db    01h, 00h, 03h, 00h ; actl regs
3308 00006FDD 0000000000300F0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 30h, 0Fh, 0Fh, 0FFh ; grdc regs
3309                              <1> vga_mode_06h:      ; 640*200 graphics, 2 colors, CGA
3310                              <1>       ; 11/11/2020
3311 00006FE6 5018080040          <1>       db    80, 24, 8, 00h, 40h   ;   tw, th-1, ch, slength
3312 00006FEB 01010006            <1>       db    01h, 01h, 00h, 06h ; sequ regs
3313 00006FEF 63                  <1>       db    63h   ; misc reg
3314 00006FF0 5F4F50825480BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
3315 00006FF8 00C1000000000000    <1>       db    00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
3316 00007000 9C8E8F280096B9C2    <1>       db    9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0C2h
3317 00007008 FF                  <1>       db    0FFh  ; crtc regs
3318 00007009 0017171717171717    <1>       db    00h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
3319 00007011 1717171717171717    <1>       db    17h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
3320 00007019 01000100            <1>       db    01h, 00h, 01, 00h  ; actl regs
3321 0000701D 0000000000000D0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 0Dh, 0Fh, 0FFh ; grdc regs
3322                              <1> vga_mode_13h:  ; mode 13h, 300*200, 256 colors, linear
3323                              <1>       ; 11/11/2020
3324                              <1>       ;db   40, 24, 8, 00h, 20h ; tw, th-1, ch, slength (5)
3325                              <1>       ; 23/11/2020 - 10/11/2020
3326 00007026 28180800FA          <1>       db    40, 24, 8, 00h, 0FAh ; tw, th-1, ch, slength (5)
3327 0000702B 010F000E            <1>       db    01h, 0Fh, 00h, 0Eh ; sequ regs (4)
3328 0000702F 63                  <1>       db    63h   ; misc reg (1)
3329 00007030 5F4F50825480BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
3330 00007038 0041000000000000    <1>       db    00h, 041h, 00h, 00h, 00h, 00h, 00h, 00h
3331 00007040 9C8E8F284096B9A3    <1>       db    9Ch, 8Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h
3332 00007048 FF                  <1>       db    0FFh  ; crtc regs (25)
3333 00007049 0001020304050607    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
3334 00007051 08090A0B0C0D0E0F    <1>       db    08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
3335 00007059 41000F00            <1>       db    41h, 00h, 0Fh, 00h  ; actl regs (20)
3336                              <1>       ; 10/11/2020
3337                              <1>       ;db   41h, 01h, 0Fh, 13h  ; actl regs (20)
3338 0000705D 000000000040050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs (9)
3339                              <1> vga_mode_setl equ $ - vga_mode_13h  ; = 64
3340                              <1> vga_mode_F0h:  ; mode X ; 320*240, 256 colors, planar
3341 00007066 2818080000          <1>       db    40, 24, 8, 00h, 00h ; tw, th-1, ch, slength
3342 0000706B 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
3343 0000706F E3                  <1>       db    0E3h  ; misc reg
3344 00007070 5F4F508254800D3E    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Dh, 3Eh
3345 00007078 0041000000000000    <1>       db    00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
3346 00007080 EAACDF2800E706E3    <1>       db    0EAh, 0ACh, 0DFh, 28h, 00h, 0E7h, 06h, 0E3h
3347 00007088 FF                  <1>       db    0FFh  ; crtc regs (25)
3348 00007089 0001020304050607    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
3349 00007091 08090A0B0C0D0E0F    <1>       db    08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
3350 00007099 41000F00            <1>       db    41h, 00h, 0Fh, 00h  ; actl regs
3351 0000709D 000000000040050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs
3352                              <1> vga_mode_12h:  ; mode 12h, 640*480, 16 colors, planar
3353                              <1>       ; 11/11/2020
3354                              <1>       ;db   80, 29, 16, 0, 0 ; tw, th-1, ch, slength
3355                              <1>       ; 09/11/2020
3356 000070A6 501D1000A0          <1>       db    80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
3357 000070AB 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
3358 000070AF E3                  <1>       db    0E3h  ; misc reg
3359 000070B0 5F4F508254800B3E    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
3360                              <1>       ; 09/11/2020
```

```
                                <1>       ;db    5Fh, 4Fh, 50h, 82h, 53h, 9Fh, 0Bh, 3Eh
000070B8 0040000000000000       <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
000070C0 EA8CDF2800E704E3       <1>       db     0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
                                <1>       ; 09/11/2020
                                <1>       ;db    0E9h, 8Bh, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
000070C8 FF                     <1>       db     0FFh   ; crtc regs
000070C9 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
000070D1 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
000070D9 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
000070DD 000000000000050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_6Ah:  ; mode 6Ah, 800*600, 16 colors, planar
                                <1>       ; 11/11/2020
000070E6 6424100000             <1>       db     100, 36, 16, 00h, 00h ; tw, th-1, ch, slength
000070EB 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000070EF E3                     <1>       db     0E3h   ; misc reg
000070F0 7F6363836B1B72F0       <1>       db     7Fh, 63h, 63h, 83h, 6Bh, 1Bh, 72h, 0F0h
000070F8 0060000000000000       <1>       db     00h, 60h, 00h, 00h, 00h, 00h, 00h, 00h
00007100 598D5732005773E3       <1>       db     59h, 8Dh, 57h, 32h, 00h, 57h, 73h, 0E3h
00007108 FF                     <1>       db     0FFh   ; crtc regs
00007109 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00007111 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
00007119 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000711D 000000000000050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_0Dh:  ; mode 0Dh, 320*200, 16 colors, planar
00007126 2818080020             <1>       db     40, 24, 8, 00h, 20h ; tw, th-1, ch, slength
0000712B 090F0006               <1>       db     09h, 0Fh, 00h, 06h ; sequ regs
0000712F 63                     <1>       db     63h    ; misc reg
00007130 2D2728902B80BF1F       <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
00007138 00C0000000000000       <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
00007140 9C8E8F140096B9E3       <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0E3h
00007148 FF                     <1>       db     0FFh   ; crtc regs
00007149 0001020304050607       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00007151 1011121314151617       <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
00007159 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000715D 000000000000050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_0Eh:  ; mode 0Eh, 640*200, 16 colors, planar
00007166 5018080040             <1>       db     80, 24, 8, 00h, 40h ; tw, th-1, ch, slength
0000716B 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
0000716F 63                     <1>       db     63h    ; misc reg
00007170 5F4F50825480BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00007178 00C0000000000000       <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
00007180 9C8E8F280096B9E3       <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0E3h
00007188 FF                     <1>       db     0FFh   ; crtc regs
00007189 0001020304050607       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00007191 1011121314151617       <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
00007199 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000719D 000000000000050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_10h: ; mode 10h, 640*350, 16 colors, planar
000071A6 50180E0080             <1>       db     80, 24, 14, 00h, 80h ; tw, th-1, ch, slength
000071AB 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000071AF A3                     <1>       db     0A3h   ; misc reg
000071B0 5F4F50825480BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
000071B8 0040000000000000       <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
000071C0 83855D280F63BAE3       <1>       db     83h, 85h, 5Dh, 28h, 0Fh, 63h, 0BAh, 0E3h
000071C8 FF                     <1>       db     0FFh   ; crtc regs
000071C9 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
000071D1 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
000071D9 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
000071DD 000000000000050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_11h: ; mode 11h, 640*480, mono color, planar
                                <1>       ; 11/11/2020
000071E6 501D1000A0             <1>       db     80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
000071EB 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000071EF E3                     <1>       db     0E3h   ; misc reg
000071F0 5F4F508254800B3E       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
000071F8 0040000000000000       <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00007200 EA8CDF2800E704C3       <1>       db     0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0C3h ; 11/11/2020
00007208 FF                     <1>       db     0FFh   ; crtc regs
00007209 003F003F003F003F       <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00007211 003F003F003F003F       <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00007219 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000721D 000000000000050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
                                <1> end_of_vga_params:
                                <1>
                                <1> ; /// End Of VIDEO DATA ///
                                <1>
                                <1> ; 23/11/2020
                                <1> ; VBE 2 BOCHS/QEMU emulator extensions
                                <1> ;     for TRDOS 386 v2 kernel (video bios)
                                <1>
                                <1> ; vbetables.h by Volker Rupper (02/01/2020)
                                <1>
                                <1> b_vbe_modes:
                                <1>       ;/* standard VESA modes */
00007226 0001800290010800       <1>       dw 100h,  640, 400,  8
0000722E 01018002E0010800       <1>       dw 101h,  640, 480,  8
00007236 0301200358020800       <1>       dw 103h,  800, 600,  8
0000723E 0501000400030800       <1>       dw 105h, 1024, 768,  8
00007246 0E014001C8001000       <1>       dw 10Eh,  320, 200, 16
0000724E 0F014001C8001800       <1>       dw 10Fh,  320, 200, 24
00007256 11018002E0011000       <1>       dw 111h,  640, 480, 16
0000725E 12018002E0011800       <1>       dw 112h,  640, 480, 24
00007266 1401200358021000       <1>       dw 114h,  800, 600, 16
0000726E 1501200358021800       <1>       dw 115h,  800, 600, 24
00007276 1701000400031000       <1>       dw 117h, 1024, 768, 16
0000727E 1801000400031800       <1>       dw 118h, 1024, 768, 24
                                <1>
                                <1> ;/* BOCHS/PLEX86 'own' mode numbers */
00007286 40014001C8002000       <1>       dw 140h,  320,  200, 32
0000728E 4101800290012000       <1>       dw 141h,  640,  400, 32
00007296 42018002E0012000       <1>       dw 142h,  640,  480, 32
0000729E 4301200358022000       <1>       dw 143h,  800,  600, 32
000072A6 4401000400032000       <1>       dw 144h, 1024,  768, 32
000072AE 46014001C8000800       <1>       dw 146h,  320,  200,  8
000072B6 8D010005D0021000       <1>       dw 18Dh, 1280,  720, 16
```

```
3466 000072BE 8E010005D0021800    <1>       dw 18Eh, 1280,  720, 24
3467 000072C6 8F010005D0022000    <1>       dw 18Fh, 1280,  720, 32
3468 000072CE 9001800738041000    <1>       dw 190h, 1920, 1080, 16
3469 000072D6 9101800738041800    <1>       dw 191h, 1920, 1080, 24
3470 000072DE 9201800738042000    <1>       dw 192h, 1920, 1080, 32
3471                              <1>
3472                              <1> end_of_b_vbe_modes:
3473                              <1>
3474                              <1> MA1 equ VBE_MODE_ATTRIBUTE_SUPPORTED
3475                              <1> MA2 equ VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE
3476                              <1> MA3 equ VBE_MODE_ATTRIBUTE_COLOR_MODE
3477                              <1> MA4 equ VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE
3478                              <1> MA5 equ      VBE_MODE_ATTRIBUTE_GRAPHICS_MODE
3479                              <1>
3480                              <1> MODE_ATTRIBUTES equ MA1|MA2|MA3|MA4|MA5
3481                              <1>
3482                              <1> WA1 equ VBE_WINDOW_ATTRIBUTE_RELOCATABLE
3483                              <1> WA2 equ      VBE_WINDOW_ATTRIBUTE_READABLE
3484                              <1> WA3 equ VBE_WINDOW_ATTRIBUTE_WRITEABLE
3485                              <1>
3486                              <1> WINA_ATTRIBUTES equ WA1|WA2|WA3
3487                              <1>
3488                              <1> ; 24/11/2020
3489                              <1>
3490                              <1> %if 0
3491                              <1>
3492                              <1> MODE_INFO_LIST:
3493                              <1>
3494                              <1> ; 24/11/2020
3495                              <1> ; '%if 0' disables 24 mode info tables here, until %endif
3496                              <1> ; ('set_mode_info_list' will set only 1 list for selected mode)
3497                              <1> ; (Purpose: To save about 1 KB kernel size by removing fixed data)
3498                              <1>
3499                              <1>                     dw     0100h ; 640x400x8
3500                              <1> ModeAttributes1:    dw     MODE_ATTRIBUTES
3501                              <1> WinAAttributes1:    db     WINA_ATTRIBUTES
3502                              <1> WinBAttributes1:    db     0
3503                              <1> WinGranularity1:    dw     VBE_DISPI_BANK_SIZE_KB
3504                              <1> WinSize1:           dw     VBE_DISPI_BANK_SIZE_KB
3505                              <1> WinASegment1:               dw     VGAMEM_GRAPH
3506                              <1> WinBSegment1:               dw     0000h
3507                              <1> WinFuncPtr1:        dd     0
3508                              <1> BytesPerScanLine1:          dw     640
3509                              <1> XResolution1:               dw     640
3510                              <1> YResolution1:               dw     400
3511                              <1> XCharSize1:         db     8
3512                              <1> YCharSize1:         db     16
3513                              <1> NumberOfPlanes1:    db     1
3514                              <1> BitsPerPixel1:              db     8
3515                              <1> NumberOfBanks1:             db     4
3516                              <1> MemoryModel1:               db     VBE_MEMORYMODEL_PACKED_PIXEL
3517                              <1> BankSize1:          db     0
3518                              <1> NumberOfImagePages1:        db     64
3519                              <1> Reserved_page1:             db     0
3520                              <1> RedMaskSize1:               db     0
3521                              <1> RedFieldPosition1:          db     0
3522                              <1> GreenMaskSize1:             db     0
3523                              <1> GreenFieldPosition1:        db     0
3524                              <1> BlueMaskSize1:              db     0
3525                              <1> BlueFieldPosition1:         db     0
3526                              <1> RsvdMaskSize1:              db     0
3527                              <1> RsvdFieldPosition1:         db     0
3528                              <1> DirectColorModeInfo1:       db     0
3529                              <1> PhysBasePtr1:               dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
3530                              <1> OffScreenMemOffset1:        dd     0
3531                              <1> OffScreenMemSize1:          dw     0
3532                              <1> LinBytesPerScanLine1:       dw     640
3533                              <1> BnkNumberOfPages1:          db     0
3534                              <1> LinNumberOfPages1:          db     0
3535                              <1> LinRedMaskSize1:    db     0
3536                              <1> LinRedFieldPosition1:       db     0
3537                              <1> LinGreenMaskSize1:          db     0
3538                              <1> LinGreenFieldPosition1: db      0
3539                              <1> LinBlueMaskSize1:  db     0
3540                              <1> LinBlueFieldPosition1:    db     0
3541                              <1> LinRsvdMaskSize1:  db     0
3542                              <1> LinRsvdFieldPosition1:    db     0
3543                              <1> MaxPixelClock1:           dd     0
3544                              <1>
3545                              <1>                     dw     0101h ; 640x480x8
3546                              <1> ModeAttributes2:    dw     MODE_ATTRIBUTES
3547                              <1> WinAAttributes2:    db     WINA_ATTRIBUTES
3548                              <1> WinBAttributes2:    db     0
3549                              <1> WinGranularity2:    dw     VBE_DISPI_BANK_SIZE_KB
3550                              <1> WinSize2:           dw     VBE_DISPI_BANK_SIZE_KB
3551                              <1> WinASegment2:               dw     VGAMEM_GRAPH
3552                              <1> WinBSegment2:               dw     0000h
3553                              <1> WinFuncPtr2:        dd     0
3554                              <1> BytesPerScanLine2: dw     640
3555                              <1> XResolution2:               dw     640
3556                              <1> YResolution2:               dw     480
3557                              <1> XCharSize2:         db     8
3558                              <1> YCharSize2:         db     16
3559                              <1> NumberOfPlanes2:    db     1
3560                              <1> BitsPerPixel2:              db     8
3561                              <1> NumberOfBanks2:             db     5
3562                              <1> MemoryModel2:               db     VBE_MEMORYMODEL_PACKED_PIXEL
3563                              <1> BankSize2:          db     0
3564                              <1> NumberOfImagePages2:        db     53
3565                              <1> Reserved_page2:             db     0
3566                              <1> RedMaskSize2:               db     0
3567                              <1> RedFieldPosition2: db     0
3568                              <1> GreenMaskSize2:             db     0
3569                              <1> GreenFieldPosition2:        db     0
3570                              <1> BlueMaskSize2:              db     0
```

```
3571                            <1> BlueFieldPosition2:       db    0
3572                            <1> RsvdMaskSize2:            db    0
3573                            <1> RsvdFieldPosition2:       db    0
3574                            <1> DirectColorModeInfo2:     db    0
3575                            <1> PhysBasePtr2:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
3576                            <1> OffScreenMemOffset2:      dd    0
3577                            <1> OffScreenMemSize2: dw     0
3578                            <1> LinBytesPerScanLine2:     dw    640
3579                            <1> BnkNumberOfPages2:        db    0
3580                            <1> LinNumberOfPages2:        db    0
3581                            <1> LinRedMaskSize2:   db     0
3582                            <1> LinRedFieldPosition2:     db    0
3583                            <1> LinGreenMaskSize2:        db    0
3584                            <1> LinGreenFieldPosition2: db      0
3585                            <1> LinBlueMaskSize2:  db     0
3586                            <1> LinBlueFieldPosition2:    db    0
3587                            <1> LinRsvdMaskSize2:  db     0
3588                            <1> LinRsvdFieldPosition2:    db    0
3589                            <1> MaxPixelClock2:           dd    0
3590                            <1>
3591                            <1>                     dw    0103h ; 800x600x8
3592                            <1> ModeAttributes3:    dw    MODE_ATTRIBUTES
3593                            <1> WinAAttributes3:    db    WINA_ATTRIBUTES
3594                            <1> WinBAttributes3:    db    0
3595                            <1> WinGranularity3:    dw    VBE_DISPI_BANK_SIZE_KB
3596                            <1> WinSize3:           dw    VBE_DISPI_BANK_SIZE_KB
3597                            <1> WinASegment3:             dw    VGAMEM_GRAPH
3598                            <1> WinBSegment3:             dw    0000h
3599                            <1> WinFuncPtr3:        dd    0
3600                            <1> BytesPerScanLine3: dw     800
3601                            <1> XResolution3:             dw    800
3602                            <1> YResolution3:             dw    600
3603                            <1> XCharSize3:         db    8
3604                            <1> YCharSize3:         db    16
3605                            <1> NumberOfPlanes3:    db    1
3606                            <1> BitsPerPixel3:            db    8
3607                            <1> NumberOfBanks3:           db    8
3608                            <1> MemoryModel3:             db    VBE_MEMORYMODEL_PACKED_PIXEL
3609                            <1> BankSize3:          db    0
3610                            <1> NumberOfImagePages3:      db    33
3611                            <1> Reserved_page3:           db    0
3612                            <1> RedMaskSize3:             db    0
3613                            <1> RedFieldPosition3: db     0
3614                            <1> GreenMaskSize3:           db    0
3615                            <1> GreenFieldPosition3:      db    0
3616                            <1> BlueMaskSize3:            db    0
3617                            <1> BlueFieldPosition3:       db    0
3618                            <1> RsvdMaskSize3:            db    0
3619                            <1> RsvdFieldPosition3:       db    0
3620                            <1> DirectColorModeInfo3:     db    0
3621                            <1> PhysBasePtr3:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
3622                            <1> OffScreenMemOffset3:      dd    0
3623                            <1> OffScreenMemSize3: dw     0
3624                            <1> LinBytesPerScanLine3:     dw    800
3625                            <1> BnkNumberOfPages3:        db    0
3626                            <1> LinNumberOfPages3:        db    0
3627                            <1> LinRedMaskSize3:   db     0
3628                            <1> LinRedFieldPosition3:     db    0
3629                            <1> LinGreenMaskSize3:        db    0
3630                            <1> LinGreenFieldPosition:    db    0
3631                            <1> LinBlueMaskSize:   db     0
3632                            <1> LinBlueFieldPosition:     db    0
3633                            <1> LinRsvdMaskSize:   db     0
3634                            <1> LinRsvdFieldPosition:     db    0
3635                            <1> MaxPixelClock:            dd    0
3636                            <1>
3637                            <1>                     dw    0105h ; 1024x768x8
3638                            <1> ModeAttributes4:    dw    MODE_ATTRIBUTES
3639                            <1> WinAAttributes4:    db    WINA_ATTRIBUTES
3640                            <1> WinBAttributes4:    db    0
3641                            <1> WinGranularity4:    dw    VBE_DISPI_BANK_SIZE_KB
3642                            <1> WinSize4:           dw    VBE_DISPI_BANK_SIZE_KB
3643                            <1> WinASegment4:             dw    VGAMEM_GRAPH
3644                            <1> WinBSegment4:             dw    0000h
3645                            <1> WinFuncPtr4:        dd    0
3646                            <1> BytesPerScanLine4: dw     1024
3647                            <1> XResolution4:             dw    1024
3648                            <1> YResolution4:             dw    768
3649                            <1> XCharSize4:         db    8
3650                            <1> YCharSize4:         db    16
3651                            <1> NumberOfPlanes4:    db    1
3652                            <1> BitsPerPixel4:            db    8
3653                            <1> NumberOfBanks4:           db    12
3654                            <1> MemoryModel4:             db    VBE_MEMORYMODEL_PACKED_PIXEL
3655                            <1> BankSize4:          db    0
3656                            <1> NumberOfImagePages4:      db    20
3657                            <1> Reserved_page4:           db    0
3658                            <1> RedMaskSize4:             db    0
3659                            <1> RedFieldPosition4: db     0
3660                            <1> GreenMaskSize4:           db    0
3661                            <1> GreenFieldPosition4:      db    0
3662                            <1> BlueMaskSize4:            db    0
3663                            <1> BlueFieldPosition4:       db    0
3664                            <1> RsvdMaskSize4:            db    0
3665                            <1> RsvdFieldPosition4:       db    0
3666                            <1> DirectColorModeInfo4:     db    0
3667                            <1> PhysBasePtr4:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
3668                            <1> OffScreenMemOffset4:      dd    0
3669                            <1> OffScreenMemSize4: dw     0
3670                            <1> LinBytesPerScanLine4:     dw    1024
3671                            <1> BnkNumberOfPages4:        db    0
3672                            <1> LinNumberOfPages4:        db    0
3673                            <1> LinRedMaskSize4:   db     0
3674                            <1> LinRedFieldPosition4:     db    0
3675                            <1> LinGreenMaskSize4:        db    0
```

```
3676                            <1> LinGreenFieldPosition4: db      0
3677                            <1> LinBlueMaskSize4:  db     0
3678                            <1> LinBlueFieldPosition4:    db     0
3679                            <1> LinRsvdMaskSize4:  db     0
3680                            <1> LinRsvdFieldPosition4:    db     0
3681                            <1> MaxPixelClock4:           dd     0
3682                            <1>
3683                            <1>                     dw     010Eh ; 320x200x16
3684                            <1> ModeAttributes5:    dw     MODE_ATTRIBUTES
3685                            <1> WinAAttributes5:    db     WINA_ATTRIBUTES
3686                            <1> WinBAttributes5:    db     0
3687                            <1> WinGranularity5:    dw     VBE_DISPI_BANK_SIZE_KB
3688                            <1> WinSize5:           dw     VBE_DISPI_BANK_SIZE_KB
3689                            <1> WinASegment5:             dw     VGAMEM_GRAPH
3690                            <1> WinBSegment5:             dw     0000h
3691                            <1> WinFuncPtr5:        dd     0
3692                            <1> BytesPerScanLine5: dw     640
3693                            <1> XResolution5:             dw     320
3694                            <1> YResolution5:             dw     200
3695                            <1> XCharSize5:         db     8
3696                            <1> YCharSize5:         db     16
3697                            <1> NumberOfPlanes5:    db     1
3698                            <1> BitsPerPixel5:            db     16
3699                            <1> NumberOfBanks5:           db     2
3700                            <1> MemoryModel5:             db     VBE_MEMORYMODEL_DIRECT_COLOR
3701                            <1> BankSize5:          db     0
3702                            <1> NumberOfImagePages5:      db     130
3703                            <1> Reserved_page5:           db     0
3704                            <1> RedMaskSize5:             db     5
3705                            <1> RedFieldPosition5: db     11
3706                            <1> GreenMaskSize5:           db     6
3707                            <1> GreenFieldPosition5:      db     5
3708                            <1> BlueMaskSize5:            db     5
3709                            <1> BlueFieldPosition5:       db     0
3710                            <1> RsvdMaskSize5:            db     0
3711                            <1> RsvdFieldPosition5:       db     0
3712                            <1> DirectColorModeInfo5:     db     0
3713                            <1> PhysBasePtr5:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
3714                            <1> OffScreenMemOffset5:      dd     0
3715                            <1> OffScreenMemSize5: dw     0
3716                            <1> LinBytesPerScanLine5:     dw     640
3717                            <1> BnkNumberOfPages5:        db     0
3718                            <1> LinNumberOfPages5:        db     0
3719                            <1> LinRedMaskSize5:   db     5
3720                            <1> LinRedFieldPosition5:     db     11
3721                            <1> LinGreenMaskSize5: db     6
3722                            <1> LinGreenFieldPosition5:   db     5
3723                            <1> LinBlueMaskSize5:  db     5
3724                            <1> LinBlueFieldPosition5:    db     0
3725                            <1> LinRsvdMaskSize5:  db     0
3726                            <1> LinRsvdFieldPosition5:    db     0
3727                            <1> MaxPixelClock5:           dd     0
3728                            <1>
3729                            <1>                     dw     010Fh ; 320x200x24
3730                            <1> ModeAttributes6:    dw     MODE_ATTRIBUTES
3731                            <1> WinAAttributes6:    db     WINA_ATTRIBUTES
3732                            <1> WinBAttributes6:    db     0
3733                            <1> WinGranularity6:    dw     VBE_DISPI_BANK_SIZE_KB
3734                            <1> WinSize6:           dw     VBE_DISPI_BANK_SIZE_KB
3735                            <1> WinASegment6:             dw     VGAMEM_GRAPH
3736                            <1> WinBSegment6:             dw     0000h
3737                            <1> WinFuncPtr6:        dd     0
3738                            <1> BytesPerScanLine6: dw     960
3739                            <1> XResolution6:             dw     320
3740                            <1> YResolution6:             dw     200
3741                            <1> XCharSize6:         db     8
3742                            <1> YCharSize6:         db     16
3743                            <1> NumberOfPlanes6:    db     1
3744                            <1> BitsPerPixel6:            db     24
3745                            <1> NumberOfBanks6:           db     3
3746                            <1> MemoryModel6:             db     VBE_MEMORYMODEL_DIRECT_COLOR
3747                            <1> BankSize6:          db     0
3748                            <1> NumberOfImagePages6:      db     86
3749                            <1> Reserved_page6:           db     0
3750                            <1> RedMaskSize6:             db     8
3751                            <1> RedFieldPosition6: db     16
3752                            <1> GreenMaskSize6:           db     8
3753                            <1> GreenFieldPosition6:      db     8
3754                            <1> BlueMaskSize6:            db     8
3755                            <1> BlueFieldPosition6:       db     0
3756                            <1> RsvdMaskSize6:            db     0
3757                            <1> RsvdFieldPosition6:       db     0
3758                            <1> DirectColorModeInfo6:     db     0
3759                            <1> PhysBasePtr6:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
3760                            <1> OffScreenMemOffset6:      dd     0
3761                            <1> OffScreenMemSize6: dw     0
3762                            <1> LinBytesPerScanLine6:     dw     960
3763                            <1> BnkNumberOfPages6:        db     0
3764                            <1> LinNumberOfPages6:        db     0
3765                            <1> LinRedMaskSize6:   db     8
3766                            <1> LinRedFieldPosition6:     db     16
3767                            <1> LinGreenMaskSize6: db     8
3768                            <1> LinGreenFieldPosition6:   db     8
3769                            <1> LinBlueMaskSize6:  db     8
3770                            <1> LinBlueFieldPosition6:    db     0
3771                            <1> LinRsvdMaskSize6:  db     0
3772                            <1> LinRsvdFieldPosition6:    db     0
3773                            <1> MaxPixelClock6:           dd     0
3774                            <1>
3775                            <1>                     dw     0111h ; 640x480x16
3776                            <1> ModeAttributes7:    dw     MODE_ATTRIBUTES
3777                            <1> WinAAttributes7:    db     WINA_ATTRIBUTES
3778                            <1> WinBAttributes7:    db     0
3779                            <1> WinGranularity7:    dw     VBE_DISPI_BANK_SIZE_KB
3780                            <1> WinSize7:           dw     VBE_DISPI_BANK_SIZE_KB
```

```
                        <1> WinASegment7:             dw    VGAMEM_GRAPH
                        <1> WinBSegment7:             dw    0000h
                        <1> WinFuncPtr7:       dd     0
                        <1> BytesPerScanLine7: dw     1280
                        <1> XResolution7:             dw    640
                        <1> YResolution7:             dw    480
                        <1> XCharSize7:        db     8
                        <1> YCharSize7:        db     16
                        <1> NumberOfPlanes7:   db     1
                        <1> BitsPerPixel7:            db    16
                        <1> NumberOfBanks7:           db    10
                        <1> MemoryModel7:             db    VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize7:         db     0
                        <1> NumberOfImagePages7:      db    26
                        <1> Reserved_page7:           db    0
                        <1> RedMaskSize7:             db    5
                        <1> RedFieldPosition7: db     11
                        <1> GreenMaskSize7:           db    6
                        <1> GreenFieldPosition7:      db    5
                        <1> BlueMaskSize7:            db    5
                        <1> BlueFieldPosition7:       db    0
                        <1> RsvdMaskSize7:            db    0
                        <1> RsvdFieldPosition7:       db    0
                        <1> DirectColorModeInfo7:     db    0
                        <1> PhysBasePtr7:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS,
                        <1> OffScreenMemOffset7:      dd    0
                        <1> OffScreenMemSize7: dw     0
                        <1> LinBytesPerScanLine7:     dw    1280
                        <1> BnkNumberOfPages7:        db    0
                        <1> LinNumberOfPages7:        db    0
                        <1> LinRedMaskSize7:   db     5
                        <1> LinRedFieldPosition7:     db    11
                        <1> LinGreenMaskSize7: db     6
                        <1> LinGreenFieldPosition7:   db    5
                        <1> LinBlueMaskSize7:  db     5
                        <1> LinBlueFieldPosition7:    db    0
                        <1> LinRsvdMaskSize7:  db     0
                        <1> LinRsvdFieldPosition7:    db    0
                        <1> MaxPixelClock7:           dd    0
                        <1>
                        <1>                     dw    0112h ; 640x480x24
                        <1> ModeAttributes8:    dw    MODE_ATTRIBUTES
                        <1> WinAAttributes8:    db    WINA_ATTRIBUTES
                        <1> WinBAttributes8:    db    0
                        <1> WinGranularity8:    dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize8:           dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment8:             dw    VGAMEM_GRAPH
                        <1> WinBSegment8:             dw    0000h
                        <1> WinFuncPtr8:       dd     0
                        <1> BytesPerScanLine8: dw     1920
                        <1> XResolution8:             dw    640
                        <1> YResolution8:             dw    480
                        <1> XCharSize8:        db     8
                        <1> YCharSize8:        db     16
                        <1> NumberOfPlanes8:   db     1
                        <1> BitsPerPixel8:            db    24
                        <1> NumberOfBanks8:           db    15
                        <1> MemoryModel8:             db    VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize8:         db     0
                        <1> NumberOfImagePages8:      db    17
                        <1> Reserved_page8:           db    0
                        <1> RedMaskSize8:             db    8
                        <1> RedFieldPosition8: db     16
                        <1> GreenMaskSize8:           db    8
                        <1> GreenFieldPosition8:      db    8
                        <1> BlueMaskSize8:            db    8
                        <1> BlueFieldPosition8:       db    0
                        <1> RsvdMaskSize8:            db    0
                        <1> RsvdFieldPosition8:       db    0
                        <1> DirectColorModeInfo8:     db    0
                        <1> PhysBasePtr8:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset8:      dd    0
                        <1> OffScreenMemSize8: dw     0
                        <1> LinBytesPerScanLine8:     dw    1920
                        <1> BnkNumberOfPages8:        db    0
                        <1> LinNumberOfPages8:        db    0
                        <1> LinRedMaskSize8:   db     8
                        <1> LinRedFieldPosition8:     db    16
                        <1> LinGreenMaskSize8: db     8
                        <1> LinGreenFieldPosition8:   db    8
                        <1> LinBlueMaskSize8:  db     8
                        <1> LinBlueFieldPosition8:    db    0
                        <1> LinRsvdMaskSize8:  db     0
                        <1> LinRsvdFieldPosition8:    db    0
                        <1> MaxPixelClock8:           dd    0
                        <1>
                        <1>                     dw    0114h ; 800x600x16
                        <1> ModeAttributes9:    dw    MODE_ATTRIBUTES
                        <1> WinAAttributes9:    db    WINA_ATTRIBUTES
                        <1> WinBAttributes9:    db    0
                        <1> WinGranularity9:    dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize9:           dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment9:             dw    VGAMEM_GRAPH
                        <1> WinBSegment9:             dw    0000h
                        <1> WinFuncPtr9:       dd     0
                        <1> BytesPerScanLine9: dw     1600
                        <1> XResolution9:             dw    800
                        <1> YResolution9:             dw    600
                        <1> XCharSize9:        db     8
                        <1> YCharSize9:        db     16
                        <1> NumberOfPlanes9:   db     1
                        <1> BitsPerPixel9:            db    16
                        <1> NumberOfBanks9:           db    15
                        <1> MemoryModel9:             db    VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize9:         db     0
```

```
3886                            <1> NumberOfImagePages9:      db    16
3887                            <1> Reserved_page9:           db    0
3888                            <1> RedMaskSize9:             db    5
3889                            <1> RedFieldPosition9: db     11
3890                            <1> GreenMaskSize9:           db    6
3891                            <1> GreenFieldPosition9:      db    5
3892                            <1> BlueMaskSize9:            db    5
3893                            <1> BlueFieldPosition9:       db    0
3894                            <1> RsvdMaskSize9:            db    0
3895                            <1> RsvdFieldPosition9:       db    0
3896                            <1> DirectColorModeInfo9:     db    0
3897                            <1> PhysBasePtr9:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
3898                            <1> OffScreenMemOffset9:      dd    0
3899                            <1> OffScreenMemSize9: dw     0
3900                            <1> LinBytesPerScanLine9:     dw    1600
3901                            <1> BnkNumberOfPages9:        db    0
3902                            <1> LinNumberOfPages9:        db    0
3903                            <1> LinRedMaskSize9:   db     5
3904                            <1> LinRedFieldPosition9:     db    11
3905                            <1> LinGreenMaskSize9: db     6
3906                            <1> LinGreenFieldPosition9:   db    5
3907                            <1> LinBlueMaskSize9:  db     5
3908                            <1> LinBlueFieldPosition9:    db    0
3909                            <1> LinRsvdMaskSize9:  db     0
3910                            <1> LinRsvdFieldPosition9:    db    0
3911                            <1> MaxPixelClock9:           dd    0
3912                            <1>
3913                            <1>                     dw    0115h ; 800x600x24
3914                            <1> ModeAttributes10:   dw    MODE_ATTRIBUTES
3915                            <1> WinAAttributes10:   db    WINA_ATTRIBUTES
3916                            <1> WinBAttributes10:   db    0
3917                            <1> WinGranularity10:   dw    VBE_DISPI_BANK_SIZE_KB
3918                            <1> WinSize10:          dw    VBE_DISPI_BANK_SIZE_KB
3919                            <1> WinASegment10:            dw    VGAMEM_GRAPH
3920                            <1> WinBSegment10:            dw    0000h
3921                            <1> WinFuncPtr10:             dd    0
3922                            <1> BytesPerScanLine10:       dw    2400
3923                            <1> XResolution10:            dw    800
3924                            <1> YResolution10:            dw    600
3925                            <1> XCharSize10:       db     8
3926                            <1> YCharSize10:       db     16
3927                            <1> NumberOfPlanes10:  db     1
3928                            <1> BitsPerPixel10:           db    24
3929                            <1> NumberOfBanks10:   db     22
3930                            <1> MemoryModel10:            db    VBE_MEMORYMODEL_DIRECT_COLOR
3931                            <1> BankSize10:        db     0
3932                            <1> NumberOfImagePages10:     db    10
3933                            <1> Reserved_page10:   db     0
3934                            <1> RedMaskSize10:            db    8
3935                            <1> RedFieldPosition10:       db    16
3936                            <1> GreenMaskSize10:   db     8
3937                            <1> GreenFieldPosition10:     db    8
3938                            <1> BlueMaskSize10:           db    8
3939                            <1> BlueFieldPosition10:      db    0
3940                            <1> RsvdMaskSize10:           db    0
3941                            <1> RsvdFieldPosition10:      db    0
3942                            <1> DirectColorModeInfo10:    db    0
3943                            <1> PhysBasePtr10:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
3944                            <1> OffScreenMemOffset10:     dd    0
3945                            <1> OffScreenMemSize10:       dw    0
3946                            <1> LinBytesPerScanLine10:    dw    2400
3947                            <1> BnkNumberOfPages10:       db    0
3948                            <1> LinNumberOfPages10:       db    0
3949                            <1> LinRedMaskSize10:  db     8
3950                            <1> LinRedFieldPosition10:    db    16
3951                            <1> LinGreenMaskSize10:       db    8
3952                            <1> LinGreenFieldPosition10:db      8
3953                            <1> LinBlueMaskSize10: db     8
3954                            <1> LinBlueFieldPosition10:   db    0
3955                            <1> LinRsvdMaskSize10: db     0
3956                            <1> LinRsvdFieldPosition10:   db    0
3957                            <1> MaxPixelClock10:   dd     0
3958                            <1>
3959                            <1>                     dw    0117h ; 1024x768x16
3960                            <1> ModeAttributes11:   dw    MODE_ATTRIBUTES
3961                            <1> WinAAttributes11:   db    WINA_ATTRIBUTES
3962                            <1> WinBAttributes11:   db    0
3963                            <1> WinGranularity11:   dw    VBE_DISPI_BANK_SIZE_KB
3964                            <1> WinSize11:          dw    VBE_DISPI_BANK_SIZE_KB
3965                            <1> WinASegment11:            dw    VGAMEM_GRAPH
3966                            <1> WinBSegment11:            dw    0000h
3967                            <1> WinFuncPtr11:             dd    0
3968                            <1> BytesPerScanLine11:       dw    2048
3969                            <1> XResolution11:            dw    1024
3970                            <1> YResolution11:            dw    768
3971                            <1> XCharSize11:       db     8
3972                            <1> YCharSize11:       db     16
3973                            <1> NumberOfPlanes11:  db     1
3974                            <1> BitsPerPixel11:           db    16
3975                            <1> NumberOfBanks11:   db     24
3976                            <1> MemoryModel11:            db    VBE_MEMORYMODEL_DIRECT_COLOR
3977                            <1> BankSize11:        db     0
3978                            <1> NumberOfImagePages11:     db    9
3979                            <1> Reserved_page11:   db     0
3980                            <1> RedMaskSize11:            db    5
3981                            <1> RedFieldPosition11:       db    11
3982                            <1> GreenMaskSize11:   db     6
3983                            <1> GreenFieldPosition11:     db    5
3984                            <1> BlueMaskSize11:           db    5
3985                            <1> BlueFieldPosition11:      db    0
3986                            <1> RsvdMaskSize11:           db    0
3987                            <1> RsvdFieldPosition11:      db    0
3988                            <1> DirectColorModeInfo11:    db    0
3989                            <1> PhysBasePtr11:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS,
3990                            <1> OffScreenMemOffset11:     dd    0
```

```
                        <1> OffScreenMemSize11:       dw     0
                        <1> LinBytesPerScanLine11:    dw     2048
                        <1> BnkNumberOfPages11:       db     0
                        <1> LinNumberOfPages11:       db     0
                        <1> LinRedMaskSize11:  db     5
                        <1> LinRedFieldPosition11:    db     11
                        <1> LinGreenMaskSize11:       db     6
                        <1> LinGreenFieldPosition11:db      5
                        <1> LinBlueMaskSize11: db     5
                        <1> LinBlueFieldPosition11:   db     0
                        <1> LinRsvdMaskSize11: db     0
                        <1> LinRsvdFieldPosition11:   db     0
                        <1> MaxPixelClock11:   dd     0
                        <1>
                        <1>                    dw     0118h ; 1024x768x24
                        <1> ModeAttributes12:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes12:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes12:  db     0
                        <1> WinGranularity12:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize12:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment12:             dw     VGAMEM_GRAPH
                        <1> WinBSegment12:             dw     0000h
                        <1> WinFuncPtr12:              dd     0
                        <1> BytesPerScanLine12:        dw     3072
                        <1> XResolution12:             dw     1024
                        <1> YResolution12:             dw     768
                        <1> XCharSize12:       db     8
                        <1> YCharSize12:       db     16
                        <1> NumberOfPlanes12:  db     1
                        <1> BitsPerPixel12:            db     24
                        <1> NumberOfBanks12:   db     36
                        <1> MemoryModel12:             db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize12:        db     0
                        <1> NumberOfImagePages12:      db     6
                        <1> Reserved_page12:   db     0
                        <1> RedMaskSize12:             db     8
                        <1> RedFieldPosition12:        db     16
                        <1> GreenMaskSize12:   db     8
                        <1> GreenFieldPosition12:      db     8
                        <1> BlueMaskSize12:            db     8
                        <1> BlueFieldPosition12:       db     0
                        <1> RsvdMaskSize12:            db     0
                        <1> RsvdFieldPosition12:       db     0
                        <1> DirectColorModeInfo12:     db     0
                        <1> PhysBasePtr12:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <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
```

```
4096                                    <1>
4097                                    <1>                     dw      0141h ; 640x400x32
4098                                    <1> ModeAttributes14:   dw      MODE_ATTRIBUTES
4099                                    <1> WinAAttributes14:   db      WINA_ATTRIBUTES
4100                                    <1> WinBAttributes14:   db      0
4101                                    <1> WinGranularity14:   dw      VBE_DISPI_BANK_SIZE_KB
4102                                    <1> WinSize14:          dw      VBE_DISPI_BANK_SIZE_KB
4103                                    <1> WinASegment14:              dw      VGAMEM_GRAPH
4104                                    <1> WinBSegment14:              dw      0000h
4105                                    <1> WinFuncPtr14:               dd      0
4106                                    <1> BytesPerScanLine14:         dw      2560
4107                                    <1> XResolution14:              dw      640
4108                                    <1> YResolution14:              dw      400
4109                                    <1> XCharSize14:        db      8
4110                                    <1> YCharSize14:        db      16
4111                                    <1> NumberOfPlanes14:   db      1
4112                                    <1> BitsPerPixel14:             db      32
4113                                    <1> NumberOfBanks14:    db      16
4114                                    <1> MemoryModel14:              db      VBE_MEMORYMODEL_DIRECT_COLOR
4115                                    <1> BankSize14:         db      0
4116                                    <1> NumberOfImagePages14:       db      15
4117                                    <1> Reserved_page14:    db      0
4118                                    <1> RedMaskSize14:              db      8
4119                                    <1> RedFieldPosition14:         db      16
4120                                    <1> GreenMaskSize14:    db      8
4121                                    <1> GreenFieldPosition14:       db      8
4122                                    <1> BlueMaskSize14:             db      8
4123                                    <1> BlueFieldPosition14:        db      0
4124                                    <1> RsvdMaskSize14:             db      8
4125                                    <1> RsvdFieldPosition14:        db      24
4126                                    <1> DirectColorModeInfo14:      db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
4127                                    <1> PhysBasePtr14:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
4128                                    <1> OffScreenMemOffset14:       dd      0
4129                                    <1> OffScreenMemSize14:         dw      0
4130                                    <1> LinBytesPerScanLine14:      dw      2560
4131                                    <1> BnkNumberOfPages14:         db      0
4132                                    <1> LinNumberOfPages14:         db      0
4133                                    <1> LinRedMaskSize14:   db      8
4134                                    <1> LinRedFieldPosition14:      db      16
4135                                    <1> LinGreenMaskSize14:         db      8
4136                                    <1> LinGreenFieldPosition14:db          8
4137                                    <1> LinBlueMaskSize14:  db      8
4138                                    <1> LinBlueFieldPosition14:     db      0
4139                                    <1> LinRsvdMaskSize14:  db      8
4140                                    <1> LinRsvdFieldPosition14:     db      24
4141                                    <1> MaxPixelClock14:    dd      0
4142                                    <1>
4143                                    <1>                     dw      0142 ; 640x480x32
4144                                    <1> ModeAttributes15:   dw      MODE_ATTRIBUTES
4145                                    <1> WinAAttributes15:   db      WINA_ATTRIBUTES
4146                                    <1> WinBAttributes15:   db      0
4147                                    <1> WinGranularity15:   dw      VBE_DISPI_BANK_SIZE_KB
4148                                    <1> WinSize15:          dw      VBE_DISPI_BANK_SIZE_KB
4149                                    <1> WinASegment15:              dw      VGAMEM_GRAPH
4150                                    <1> WinBSegment15:              dw      0000h
4151                                    <1> WinFuncPtr15:               dd      0
4152                                    <1> BytesPerScanLine15:         dw      2560
4153                                    <1> XResolution15:              dw      640
4154                                    <1> YResolution15:              dw      480
4155                                    <1> XCharSize15:        db      8
4156                                    <1> YCharSize15:        db      16
4157                                    <1> NumberOfPlanes15:   db      1
4158                                    <1> BitsPerPixel15:             db      32
4159                                    <1> NumberOfBanks15:    db      19
4160                                    <1> MemoryModel15:              db      VBE_MEMORYMODEL_DIRECT_COLOR
4161                                    <1> BankSize15:         db      0
4162                                    <1> NumberOfImagePages15:       db      12
4163                                    <1> Reserved_page15:    db      0
4164                                    <1> RedMaskSize15:              db      8
4165                                    <1> RedFieldPosition15:         db      16
4166                                    <1> GreenMaskSize15:    db      8
4167                                    <1> GreenFieldPosition15:       db      8
4168                                    <1> BlueMaskSize15:             db      8
4169                                    <1> BlueFieldPosition15:        db      0
4170                                    <1> RsvdMaskSize15:             db      8
4171                                    <1> RsvdFieldPosition15:        db      24
4172                                    <1> DirectColorModeInfo15:      db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
4173                                    <1> PhysBasePtr15:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS,
4174                                    <1> OffScreenMemOffset15:       dd      0
4175                                    <1> OffScreenMemSize15:         dw      0
4176                                    <1> LinBytesPerScanLine15:      dw      2560
4177                                    <1> BnkNumberOfPages15:         db      0
4178                                    <1> LinNumberOfPages15:         db      0
4179                                    <1> LinRedMaskSize15:   db      8
4180                                    <1> LinRedFieldPosition15:      db      16
4181                                    <1> LinGreenMaskSize15:         db      8
4182                                    <1> LinGreenFieldPosition15:db          8
4183                                    <1> LinBlueMaskSize15:  db      8
4184                                    <1> LinBlueFieldPosition15:     db      0
4185                                    <1> LinRsvdMaskSize15:  db      8
4186                                    <1> LinRsvdFieldPosition15:     db      24
4187                                    <1> MaxPixelClock15:    dd      0
4188                                    <1>
4189                                    <1>                     dw      0143h ; 800x600x32
4190                                    <1> ModeAttributes16:   dw      MODE_ATTRIBUTES
4191                                    <1> WinAAttributes16:   db      WINA_ATTRIBUTES
4192                                    <1> WinBAttributes16:   db      0
4193                                    <1> WinGranularity16:   dw      VBE_DISPI_BANK_SIZE_KB
4194                                    <1> WinSize16:          dw      VBE_DISPI_BANK_SIZE_KB
4195                                    <1> WinASegment16:              dw      VGAMEM_GRAPH
4196                                    <1> WinBSegment16:              dw      0000h
4197                                    <1> WinFuncPtr16:               dd      0
4198                                    <1> BytesPerScanLine16:         dw      3200
4199                                    <1> XResolution16:              dw      800
4200                                    <1> YResolution16:              dw      600
```

```
4201                            <1> XCharSize16:       db     8
4202                            <1> YCharSize16:       db     16
4203                            <1> NumberOfPlanes16:  db     1
4204                            <1> BitsPerPixel16:            db     32
4205                            <1> NumberOfBanks16:   db     30
4206                            <1> MemoryModel16:             db     VBE_MEMORYMODEL_DIRECT_COLOR
4207                            <1> BankSize16:        db     0
4208                            <1> NumberOfImagePages16:      db     7
4209                            <1> Reserved_page16:   db     0
4210                            <1> RedMaskSize16:             db     8
4211                            <1> RedFieldPosition16:        db     16
4212                            <1> GreenMaskSize16:   db     8
4213                            <1> GreenFieldPosition16:      db     8
4214                            <1> BlueMaskSize16:            db     8
4215                            <1> BlueFieldPosition16:       db     0
4216                            <1> RsvdMaskSize16:            db     8
4217                            <1> RsvdFieldPosition16:       db     24
4218                            <1> DirectColorModeInfo16:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
4219                            <1> PhysBasePtr16:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS,
4220                            <1> OffScreenMemOffset16:      dd     0
4221                            <1> OffScreenMemSize16:        dw     0
4222                            <1> LinBytesPerScanLine16:     dw     3200
4223                            <1> BnkNumberOfPages16:        db     0
4224                            <1> LinNumberOfPages16:        db     0
4225                            <1> LinRedMaskSize16:  db     8
4226                            <1> LinRedFieldPosition16:     db     16
4227                            <1> LinGreenMaskSize16:        db     8
4228                            <1> LinGreenFieldPosition16:db       8
4229                            <1> LinBlueMaskSize16: db     8
4230                            <1> LinBlueFieldPosition16:    db     0
4231                            <1> LinRsvdMaskSize16: db     8
4232                            <1> LinRsvdFieldPosition16:    db     24
4233                            <1> MaxPixelClock16:   dd     0
4234                            <1>
4235                            <1>                    dw     0144h ; 1024x768x32
4236                            <1> ModeAttributes17:  dw     MODE_ATTRIBUTES
4237                            <1> WinAAttributes17:  db     WINA_ATTRIBUTES
4238                            <1> WinBAttributes17:  db     0
4239                            <1> WinGranularity17:  dw     VBE_DISPI_BANK_SIZE_KB
4240                            <1> WinSize17:         dw     VBE_DISPI_BANK_SIZE_KB
4241                            <1> WinASegment17:             dw     VGAMEM_GRAPH
4242                            <1> WinBSegment17:             dw     0000h
4243                            <1> WinFuncPtr17:              dd     0
4244                            <1> BytesPerScanLine17:        dw     4096
4245                            <1> XResolution17:             dw     1024
4246                            <1> YResolution17:             dw     768
4247                            <1> XCharSize17:       db     8
4248                            <1> YCharSize17:       db     16
4249                            <1> NumberOfPlanes17:  db     1
4250                            <1> BitsPerPixel17:            db     32
4251                            <1> NumberOfBanks17:   db     48
4252                            <1> MemoryModel17:             db     VBE_MEMORYMODEL_DIRECT_COLOR
4253                            <1> BankSize17:        db     0
4254                            <1> NumberOfImagePages17:      db     4
4255                            <1> Reserved_page17:   db     0
4256                            <1> RedMaskSize17:             db     8
4257                            <1> RedFieldPosition17:        db     16
4258                            <1> GreenMaskSize17:   db     8
4259                            <1> GreenFieldPosition17:      db     8
4260                            <1> BlueMaskSize17:            db     8
4261                            <1> BlueFieldPosition17:       db     0
4262                            <1> RsvdMaskSize17:            db     8
4263                            <1> RsvdFieldPosition17:       db     24
4264                            <1> DirectColorModeInfo17:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
4265                            <1> PhysBasePtr17:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4266                            <1> OffScreenMemOffset17:      dd     0
4267                            <1> OffScreenMemSize17:        dw     0
4268                            <1> LinBytesPerScanLine17:     dw     4096
4269                            <1> BnkNumberOfPages17:        db     0
4270                            <1> LinNumberOfPages17:        db     0
4271                            <1> LinRedMaskSize17:  db     8
4272                            <1> LinRedFieldPosition17:     db     16
4273                            <1> LinGreenMaskSize17:        db     8
4274                            <1> LinGreenFieldPosition17:db       8
4275                            <1> LinBlueMaskSize17: db     8
4276                            <1> LinBlueFieldPosition17:    db     0
4277                            <1> LinRsvdMaskSize17: db     8
4278                            <1> LinRsvdFieldPosition17:    db     24
4279                            <1> MaxPixelClock17:   dd     0
4280                            <1>
4281                            <1>                    dw     0146h ; 320x200x8
4282                            <1> ModeAttributes18:  dw     MODE_ATTRIBUTES
4283                            <1> WinAAttributes18:  db     WINA_ATTRIBUTES
4284                            <1> WinBAttributes18:  db     0
4285                            <1> WinGranularity18:  dw     VBE_DISPI_BANK_SIZE_KB
4286                            <1> WinSize18:         dw     VBE_DISPI_BANK_SIZE_KB
4287                            <1> WinASegment18:             dw     VGAMEM_GRAPH
4288                            <1> WinBSegment18:             dw     0000h
4289                            <1> WinFuncPtr18:              dd     0
4290                            <1> BytesPerScanLine18:        dw     320
4291                            <1> XResolution18:             dw     320
4292                            <1> YResolution18:             dw     200
4293                            <1> XCharSize18:       db     8
4294                            <1> YCharSize18:       db     16
4295                            <1> NumberOfPlanes18:  db     1
4296                            <1> BitsPerPixel18:            db     8
4297                            <1> NumberOfBanks18:   db     1
4298                            <1> MemoryModel18:             db     VBE_MEMORYMODEL_PACKED_PIXEL
4299                            <1> BankSize18:        db     0
4300                            <1> NumberOfImagePages18:      db     255 ; 261 in vbetables.h (03/01/2020) !
4301                            <1> Reserved_page18:   db     0
4302                            <1> RedMaskSize18:             db     0
4303                            <1> RedFieldPosition18:        db     0
4304                            <1> GreenMaskSize18:   db     0
4305                            <1> GreenFieldPosition18:      db     0
```

```
4306                            <1> BlueMaskSize18:           db    0
4307                            <1> BlueFieldPosition18:      db    0
4308                            <1> RsvdMaskSize18:           db    0
4309                            <1> RsvdFieldPosition18:      db    0
4310                            <1> DirectColorModeInfo18:    db    0
4311                            <1> PhysBasePtr18:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
4312                            <1> OffScreenMemOffset18:     dd    0
4313                            <1> OffScreenMemSize18:       dw    0
4314                            <1> LinBytesPerScanLine18:    dw    320
4315                            <1> BnkNumberOfPages18:       db    0
4316                            <1> LinNumberOfPages18:       db    0
4317                            <1> LinRedMaskSize18:  db     0
4318                            <1> LinRedFieldPosition18:    db    0
4319                            <1> LinGreenMaskSize18:       db    0
4320                            <1> LinGreenFieldPosition18:db      0
4321                            <1> LinBlueMaskSize18:        db    0
4322                            <1> LinBlueFieldPosition18: db      0
4323                            <1> LinRsvdMaskSize18:        db    0
4324                            <1> LinRsvdFieldPosition18: db      0
4325                            <1> MaxPixelClock18:   dd     0
4326                            <1>
4327                            <1>                    dw     018Dh ; 1280x720x16
4328                            <1> ModeAttributes19:  dw     MODE_ATTRIBUTES
4329                            <1> WinAAttributes19:  db     WINA_ATTRIBUTES
4330                            <1> WinBAttributes19:  db     0
4331                            <1> WinGranularity19:  dw     VBE_DISPI_BANK_SIZE_KB
4332                            <1> WinSize19:         dw     VBE_DISPI_BANK_SIZE_KB
4333                            <1> WinASegment19:             dw     VGAMEM_GRAPH
4334                            <1> WinBSegment19:             dw     0000h
4335                            <1> WinFuncPtr19:              dd     0
4336                            <1> BytesPerScanLine19:        dw     2560
4337                            <1> XResolution19:             dw     1280
4338                            <1> YResolution19:             dw     720
4339                            <1> XCharSize19:       db     8
4340                            <1> YCharSize19:       db     16
4341                            <1> NumberOfPlanes19:  db     1
4342                            <1> BitsPerPixel19:            db     16
4343                            <1> NumberOfBanks19:   db     29
4344                            <1> MemoryModel19:             db     VBE_MEMORYMODEL_DIRECT_COLOR
4345                            <1> BankSize19:        db     0
4346                            <1> NumberOfImagePages19:      db     8
4347                            <1> Reserved_page19:   db     0
4348                            <1> RedMaskSize19:             db     5
4349                            <1> RedFieldPosition19:        db     11
4350                            <1> GreenMaskSize19:   db     6
4351                            <1> GreenFieldPosition19:      db     5
4352                            <1> BlueMaskSize19:            db     5
4353                            <1> BlueFieldPosition19:       db     0
4354                            <1> RsvdMaskSize19:            db     0
4355                            <1> RsvdFieldPosition19:       db     0
4356                            <1> DirectColorModeInfo19:     db     0
4357                            <1> PhysBasePtr19:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4358                            <1> OffScreenMemOffset19:      dd     0
4359                            <1> OffScreenMemSize19:        dw     0
4360                            <1> LinBytesPerScanLine19:     dw     2560
4361                            <1> BnkNumberOfPages19:        db     0
4362                            <1> LinNumberOfPages19:        db     0
4363                            <1> LinRedMaskSize19:  db     5
4364                            <1> LinRedFieldPosition19:     db     11
4365                            <1> LinGreenMaskSize19:        db     6
4366                            <1> LinGreenFieldPosition19:db      5
4367                            <1> LinBlueMaskSize19: db     5
4368                            <1> LinBlueFieldPosition19:    db     0
4369                            <1> LinRsvdMaskSize19: db     0
4370                            <1> LinRsvdFieldPosition19:    db     0
4371                            <1> MaxPixelClock19:   dd     0
4372                            <1>
4373                            <1>                    dw     018Eh ; 1280x720x24
4374                            <1> ModeAttributes20:  dw     MODE_ATTRIBUTES
4375                            <1> WinAAttributes20:  db     WINA_ATTRIBUTES
4376                            <1> WinBAttributes20:  db     0
4377                            <1> WinGranularity20:  dw     VBE_DISPI_BANK_SIZE_KB
4378                            <1> WinSize20:         dw     VBE_DISPI_BANK_SIZE_KB
4379                            <1> WinASegment20:             dw     VGAMEM_GRAPH
4380                            <1> WinBSegment20:             dw     0000h
4381                            <1> WinFuncPtr20:              dd     0
4382                            <1> BytesPerScanLine20:        dw     3840
4383                            <1> XResolution20:             dw     1280
4384                            <1> YResolution20:             dw     720
4385                            <1> XCharSize20:       db     8
4386                            <1> YCharSize20:       db     16
4387                            <1> NumberOfPlanes20:  db     1
4388                            <1> BitsPerPixel20:            db     24
4389                            <1> NumberOfBanks20:   db     43
4390                            <1> MemoryModel20:             db     VBE_MEMORYMODEL_DIRECT_COLOR
4391                            <1> BankSize20:        db     0
4392                            <1> NumberOfImagePages20:      db     5
4393                            <1> Reserved_page20:   db     0
4394                            <1> RedMaskSize20:             db     8
4395                            <1> RedFieldPosition20:        db     16
4396                            <1> GreenMaskSize20:   db     8
4397                            <1> GreenFieldPosition20:      db     8
4398                            <1> BlueMaskSize20:            db     8
4399                            <1> BlueFieldPosition20:       db     0
4400                            <1> RsvdMaskSize20:            db     0
4401                            <1> RsvdFieldPosition20:       db     0
4402                            <1> DirectColorModeInfo20:     db     0
4403                            <1> PhysBasePtr20:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4404                            <1> OffScreenMemOffset20:      dd     0
4405                            <1> OffScreenMemSize20:        dw     0
4406                            <1> LinBytesPerScanLine20:     dw     3840
4407                            <1> BnkNumberOfPages20:        db     0
4408                            <1> LinNumberOfPages20:        db     0
4409                            <1> LinRedMaskSize20:  db     8
4410                            <1> LinRedFieldPosition20:     db     16
```

```
                        <1> LinGreenMaskSize20:       db     8
                        <1> LinGreenFieldPosition20:db       8
                        <1> LinBlueMaskSize20: db     8
                        <1> LinBlueFieldPosition20:   db     0
                        <1> LinRsvdMaskSize20: db     0
                        <1> LinRsvdFieldPosition20:   db     0
                        <1> MaxPixelClock20:   dd     0
                        <1>
                        <1>                    dw     018Fh ; 1280x720x32
                        <1> ModeAttributes21:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes21:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes21:  db     0
                        <1> WinGranularity21:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize21:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment21:             dw     VGAMEM_GRAPH
                        <1> WinBSegment21:             dw     0000h
                        <1> WinFuncPtr21:              dd     0
                        <1> BytesPerScanLine21:        dw     5120
                        <1> XResolution21:             dw     1280
                        <1> YResolution21:             dw     720
                        <1> XCharSize21:       db     8
                        <1> YCharSize21:       db     16
                        <1> NumberOfPlanes21:  db     1
                        <1> BitsPerPixel21:            db     32
                        <1> NumberOfBanks21:   db     57
                        <1> MemoryModel21:             db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize21:        db     0
                        <1> NumberOfImagePages21:      db     3
                        <1> Reserved_page21:   db     0
                        <1> RedMaskSize21:             db     8
                        <1> RedFieldPosition21:        db     16
                        <1> GreenMaskSize21:   db     8
                        <1> GreenFieldPosition21:      db     8
                        <1> BlueMaskSize21:            db     8
                        <1> BlueFieldPosition21:       db     0
                        <1> RsvdMaskSize21:            db     8
                        <1> RsvdFieldPosition21:       db     24
                        <1> DirectColorModeInfo21:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
                        <1> PhysBasePtr21:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset21:      dd     0
                        <1> OffScreenMemSize21:        dw     0
                        <1> LinBytesPerScanLine21:     dw     5120
                        <1> BnkNumberOfPages21:        db     0
                        <1> LinNumberOfPages21:        db     0
                        <1> LinRedMaskSize21:  db     8
                        <1> LinRedFieldPosition21:     db     16
                        <1> LinGreenMaskSize21:        db     8
                        <1> LinGreenFieldPosition21:db       8
                        <1> LinBlueMaskSize21: db     8
                        <1> LinBlueFieldPosition21:    db     0
                        <1> LinRsvdMaskSize21: db     8
                        <1> LinRsvdFieldPosition21:    db     24
                        <1> MaxPixelClock21:   dd     0
                        <1>
                        <1>                    dw     0190h ; 1920x1080x16
                        <1> ModeAttributes22:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes22:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes22:  db     0
                        <1> WinGranularity22:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize22:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment22:             dw     VGAMEM_GRAPH
                        <1> WinBSegment22:             dw     0000h
                        <1> WinFuncPtr22:              dd     0
                        <1> BytesPerScanLine22:        dw     3840
                        <1> XResolution22:             dw     1920
                        <1> YResolution22:             dw     1080
                        <1> XCharSize22:       db     8
                        <1> YCharSize22:       db     16
                        <1> NumberOfPlanes22:  db     1
                        <1> BitsPerPixel22:            db     16
                        <1> NumberOfBanks22:   db     64
                        <1> MemoryModel22:             db     VBE_MEMORYMODEL_DIRECT_COLOR,
                        <1> BankSize22:        db     0
                        <1> NumberOfImagePages22:      db     3
                        <1> Reserved_page22:   db     0
                        <1> RedMaskSize22:             db     5
                        <1> RedFieldPosition22:        db     11
                        <1> GreenMaskSize22:   db     6
                        <1> GreenFieldPosition22:      db     5
                        <1> BlueMaskSize22:            db     5
                        <1> BlueFieldPosition22:       db     0
                        <1> RsvdMaskSize22:            db     0
                        <1> RsvdFieldPosition22:       db     0
                        <1> DirectColorModeInfo22:     db     0
                        <1> PhysBasePtr22:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset22:      dd     0
                        <1> OffScreenMemSize22:        dw     0
                        <1> LinBytesPerScanLine22:     dw     3840
                        <1> BnkNumberOfPages22:        db     0
                        <1> LinNumberOfPages22:        db     0
                        <1> LinRedMaskSize22:  db     5
                        <1> LinRedFieldPosition22:     db     11
                        <1> LinGreenMaskSize22:        db     6
                        <1> LinGreenFieldPosition22:db       5
                        <1> LinBlueMaskSize22: db     5
                        <1> LinBlueFieldPosition22:    db     0
                        <1> LinRsvdMaskSize22: db     0
                        <1> LinRsvdFieldPosition22:    db     0
                        <1> MaxPixelClock22:   dd     0
                        <1>
                        <1>                    dw     0191h ; 1920x1080x24
                        <1> ModeAttributes23:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes23:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes23:  db     0
                        <1> WinGranularity23:  dw     VBE_DISPI_BANK_SIZE_KB
```

```
4516                                    <1> WinSize23:          dw      VBE_DISPI_BANK_SIZE_KB
4517                                    <1> WinASegment23:              dw      VGAMEM_GRAPH
4518                                    <1> WinBSegment23:              dw      0000h
4519                                    <1> WinFuncPtr23:               dd      0
4520                                    <1> BytesPerScanLine23:         dw      5760
4521                                    <1> XResolution23:              dw      1920
4522                                    <1> YResolution23:              dw      1080
4523                                    <1> XCharSize23:        db      8
4524                                    <1> YCharSize23:        db      16
4525                                    <1> NumberOfPlanes23:   db      1
4526                                    <1> BitsPerPixel23:             db      24
4527                                    <1> NumberOfBanks23:    db      95
4528                                    <1> MemoryModel23:              db      VBE_MEMORYMODEL_DIRECT_COLOR
4529                                    <1> BankSize23:         db      0
4530                                    <1> NumberOfImagePages23:       db      1
4531                                    <1> Reserved_page23:    db      0
4532                                    <1> RedMaskSize23:              db      8
4533                                    <1> RedFieldPosition23:         db      16
4534                                    <1> GreenMaskSize23:    db      8
4535                                    <1> GreenFieldPosition23:       db      8
4536                                    <1> BlueMaskSize23:             db      8
4537                                    <1> BlueFieldPosition23:        db      0
4538                                    <1> RsvdMaskSize23:             db      0
4539                                    <1> RsvdFieldPosition23:        db      0
4540                                    <1> DirectColorModeInfo23:      db      0
4541                                    <1> PhysBasePtr23:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
4542                                    <1> OffScreenMemOffset23:       dd      0
4543                                    <1> OffScreenMemSize23:         dw      0
4544                                    <1> LinBytesPerScanLine23:      dw      5760
4545                                    <1> BnkNumberOfPages23:         db      0
4546                                    <1> LinNumberOfPages23:         db      0
4547                                    <1> LinRedMaskSize23:   db      8
4548                                    <1> LinRedFieldPosition23:      db      16
4549                                    <1> LinGreenMaskSize23:         db      8
4550                                    <1> LinGreenFieldPosition23:db          8
4551                                    <1> LinBlueMaskSize23: db       8
4552                                    <1> LinBlueFieldPosition23:     db      0
4553                                    <1> LinRsvdMaskSize23: db       0
4554                                    <1> LinRsvdFieldPosition23:     db      0
4555                                    <1> MaxPixelClock23:    dd      0
4556                                    <1>
4557                                    <1>                     dw      0192h ; 1920x1080x32
4558                                    <1> ModeAttributes24:   dw      MODE_ATTRIBUTES
4559                                    <1> WinAAttributes24:   db      WINA_ATTRIBUTES
4560                                    <1> WinBAttributes24:   db      0
4561                                    <1> WinGranularity24:   dw      VBE_DISPI_BANK_SIZE_KB
4562                                    <1> WinSize24:          dw      VBE_DISPI_BANK_SIZE_KB
4563                                    <1> WinASegment24:              dw      VGAMEM_GRAPH
4564                                    <1> WinBSegment24:              dw      0000h
4565                                    <1> WinFuncPtr24:               dd      0
4566                                    <1> BytesPerScanLine24:         dw      7680
4567                                    <1> XResolution24:              dw      1920
4568                                    <1> YResolution24:              dw      1080
4569                                    <1> XCharSize24:        db      8
4570                                    <1> YCharSize24:        db      16
4571                                    <1> NumberOfPlanes24:   db      1
4572                                    <1> BitsPerPixel24:             db      32
4573                                    <1> NumberOfBanks24:    db      127
4574                                    <1> MemoryModel24:              db      VBE_MEMORYMODEL_DIRECT_COLOR
4575                                    <1> BankSize24:         db      0
4576                                    <1> NumberOfImagePages24:       db      1
4577                                    <1> Reserved_page24:    db      0
4578                                    <1> RedMaskSize24:              db      8
4579                                    <1> RedFieldPosition24:         db      16
4580                                    <1> GreenMaskSize24:    db      8
4581                                    <1> GreenFieldPosition24:       db      8
4582                                    <1> BlueMaskSize24:             db      8
4583                                    <1> BlueFieldPosition24:        db      0
4584                                    <1> RsvdMaskSize24:             db      8
4585                                    <1> RsvdFieldPosition24:        db      24
4586                                    <1> DirectColorModeInfo24:      db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
4587                                    <1> PhysBasePtr24:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
4588                                    <1> OffScreenMemOffset24:       dd      0
4589                                    <1> OffScreenMemSize24:         dw      0
4590                                    <1> LinBytesPerScanLine24:      dw      7680
4591                                    <1> BnkNumberOfPages24:         db      0
4592                                    <1> LinNumberOfPages24:         db      0
4593                                    <1> LinRedMaskSize24:   db      8
4594                                    <1> LinRedFieldPosition24:      db      16
4595                                    <1> LinGreenMaskSize24:         db      8
4596                                    <1> LinGreenFieldPosition24:db          8
4597                                    <1> LinBlueMaskSize24: db       8
4598                                    <1> LinBlueFieldPosition24:     db      0
4599                                    <1> LinRsvdMaskSize24: db       8
4600                                    <1> LinRsvdFieldPosition24:     db      24
4601                                    <1> MaxPixelClock24:    dd      0
4602                                    <1>
4603                                    <1> VBE_VESA_MODE_END_OF_LIST: dw   0
4604                                    <1>
4605                                    <1> %endif
4606                                    <1>
4607                                    <1> ; 24/11/2020
4608                                    <1>
4609                                    <1> direct_color_fields:
4610                                    <1>       ; 24/11/2020
4611                                    <1>
4612                                    <1> ; (vbetables-gen.c)
4613                                    <1> ; // Direct Color fields
4614                                    <1> ; (required for direct/6 and YUV/7 memory models)
4615                                    <1> ; switch(pm->depth) {
4616                                    <1>
4617                                    <1> ;case 8:
4618 000072E6 00                     <1> r_size_8:  db 0
4619 000072E7 00                     <1> r_pos_8:   db 0
4620 000072E8 00                     <1> g_size_8:  db 0
```

```
4621 000072E9 00                <1> g_pos_8:   db 0
4622 000072EA 00                <1> b_size_8:  db 0
4623 000072EB 00                <1> b_pos_8:   db 0
4624 000072EC 00                <1> a_size_8:  db 0
4625 000072ED 00                <1> a_pos_8:   db 0
4626                            <1>
4627                            <1> ;case 16:
4628 000072EE 05                <1> r_size_16:  db 5
4629 000072EF 0B                <1> r_pos_16:   db 11
4630 000072F0 06                <1> g_size_16:  db 6
4631 000072F1 05                <1> g_pos_16:   db 5
4632 000072F2 05                <1> b_size_16:  db 5
4633 000072F3 00                <1> b_pos_16:   db 0
4634 000072F4 00                <1> a_size_16:  db 0
4635 000072F5 00                <1> a_pos_16:   db 0
4636                            <1>
4637                            <1> ;case 24:
4638 000072F6 08                <1> r_size_24:  db 8
4639 000072F7 10                <1> r_pos_24:   db 16
4640 000072F8 08                <1> g_size_24:  db 8
4641 000072F9 08                <1> g_pos_24:   db 8
4642 000072FA 08                <1> b_size_24:  db 8
4643 000072FB 00                <1> b_pos_24:   db 0
4644 000072FC 00                <1> a_size_24:  db 0
4645 000072FD 00                <1> a_pos_24:   db 0
4646                            <1>
4647                            <1> ;case 32:
4648 000072FE 08                <1> r_size_32:  db 8
4649 000072FF 10                <1> r_pos_32:   db 16
4650 00007300 08                <1> g_size_32:  db 8
4651 00007301 08                <1> g_pos_32:   db 8
4652 00007302 08                <1> b_size_32:  db 8
4653 00007303 00                <1> b_pos_32:   db 0
4654 00007304 08                <1> a_size_32:  db 8
4655 00007305 18                <1> a_pos_32:   db 24
3077                                ;%include 'diskdata.s'    ; DISK (BIOS) DATA (initialized)
3078                                ;;;
3079
3080                                Align 2
3081
3082                                %include 'sysdefs.s' ; 24/01/2015
3083                            <1> ; ****************************************************************************
3084                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - SYSTEM DEFINITIONS : sysdefs.s
3085                            <1> ; ----------------------------------------------------------------------------
3086                            <1> ; Last Update: 31/12/2017
3087                            <1> ; ----------------------------------------------------------------------------
3088                            <1> ; Beginning: 24/01/2016
3089                            <1> ; ----------------------------------------------------------------------------
3090                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3091                            <1> ; ----------------------------------------------------------------------------
3092                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3093                            <1> ; sysdefs.inc (14/11/2015)
3094                            <1> ; ****************************************************************************
3095                            <1>
3096                            <1> ; Retro UNIX 386 v1 Kernel - SYSDEFS.INC
3097                            <1> ; Last Modification: 14/11/2015
3098                            <1> ;
3099                            <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
3100                            <1> ; (Modified from
3101                            <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
3102                            <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
3103                            <1> ;     UNIX.ASM (MASM 6.11) --> SYSDEFS.INC (NASM 2.11)
3104                            <1> ; ----------------------------------------------------------------------------
3105                            <1> ;
3106                            <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
3107                            <1> ; (Original) Source Code by Ken Thompson (1971-1972)
3108                            <1> ; <Bell Laboratories (17/3/1972)>
3109                            <1> ; <Preliminary Release of UNIX Implementation Document>
3110                            <1> ;
3111                            <1> ; ****************************************************************************
3112                            <1>
3113                            <1> nproc       equ    16  ; number of processes
3114                            <1> nfiles      equ    50
3115                            <1> ntty  equ     8   ; 8+1 -> 8 (10/05/2013)
3116                            <1> nbuf  equ    4   ; 6 ;; 21/08/2015 - 'namei' buffer problem when nbuf > 4
3117                            <1>              ; NOTE: If fd0 super block buffer addres is beyond of the 1st
3118                            <1>              ; 32K, DMA r/w routine or someting else causes a jump to
3119                            <1>              ; kernel panic routine (in 'alloc' routine, in u5.s)
3120                            <1>              ; because of invalid buffer content (r/w error).
3121                            <1>              ; When all buffers are set before the end of the 1st 32k,
3122                            <1>              ; there is no problem!? (14/11/2015)
3123                            <1>
3124                            <1> ;csgmnt     equ    2000h  ; 26/05/2013 (segment of process 1)
3125                            <1> ;core equ    0          ; 19/04/2013
3126                            <1> ;ecore      equ    32768 - 64  ; 04/06/2013 (24/05/2013)
3127                            <1>       ; (if total size of argument list and arguments is 128 bytes)
3128                            <1>       ; maximum executable file size = 32768-(64+40+128-6) = 32530 bytes
3129                            <1>       ; maximum stack size = 40 bytes (+6 bytes for 'IRET' at 32570)
3130                            <1>       ; initial value of user's stack pointer = 32768-64-128-2 = 32574
3131                            <1>       ;     (sp=32768-args_space-2 at the beginning of execution)
3132                            <1>       ; argument list offset = 32768-64-128 = 32576 (if it is 128 bytes)
3133                            <1>       ; 'u' structure offset (for the '/core' dump file) = 32704
3134                            <1>       ; '/core' dump file size = 32768 bytes
3135                            <1>
3136                            <1> ; 08/03/2014
3137                            <1> ;sdsegmnt equ      6C0h  ; 256*16 bytes (swap data segment size for 16 processes)
3138                            <1> ; 19/04/2013 Retro UNIX 8086 v1 feaure only !
3139                            <1> ;;sdsegmnt equ     740h  ; swap data segment (for user structures and registers)
3140                            <1>
3141                            <1> ; 30/08/2013
3142                            <1> time_count equ 4 ; 10 --> 4 01/02/2014
3143                            <1>
3144                            <1> ; 05/02/2014
3145                            <1> ; process status
3146                            <1> ;SFREE      equ 0
```

```
3147                                    <1> ;SRUN equ 1
3148                                    <1> ;SWAIT      equ 2
3149                                    <1> ;SZOMB      equ 3
3150                                    <1> ;SSLEEP     equ 4 ; Retro UNIX 8086 V1 extension (for sleep and wakeup)
3151                                    <1>
3152                                    <1> ; 09/03/2015
3153                                    <1> userdata equ 80000h ; user structure data address for current user ; temporary
3154                                    <1> swap_queue equ 90000h - 2000h ; swap queue address ; temporary
3155                                    <1> swap_alloc_table equ 0D0000h  ;  swap allocation table address ; temporary
3156                                    <1>
3157                                    <1> ; 17/09/2015
3158                                    <1> ESPACE equ 48 ; [u.usp] (at 'sysent') - [u.sp] value for error return
3159                                    <1>
3160                                    <1> ; 31/12/2017
3161                                    <1> ; 19/02/2017
3162                                    <1> ; 15/10/2016
3163                                    <1> ; 20/05/2016
3164                                    <1> ; 19/05/2016
3165                                    <1> ; 18/05/2016
3166                                    <1> ; 29/04/2016
3167                                    <1> ; TRDOS 386 (TRDOS v2.0) system calls - temporary List
3168                                    <1> ; 14/07/2013 - 21/09/2015 (Retro UNIX 8086 & 386 system calls)
3169                                    <1> _ver  equ 0 ; Get TRDOS version (v2.0)
3170                                    <1> _exit       equ 1
3171                                    <1> _fork       equ 2
3172                                    <1> _read       equ 3
3173                                    <1> _write      equ 4
3174                                    <1> _open equ 5
3175                                    <1> _close      equ 6
3176                                    <1> _wait       equ 7
3177                                    <1> _creat      equ 8
3178                                    <1> _rename     equ 9  ; TRDOS 386, Rename File (31/12/2017)
3179                                    <1> _delete     equ 10 ; TRDOS 386, Delete File (29/12/2017)
3180                                    <1> _exec equ 11
3181                                    <1> _chdir      equ 12
3182                                    <1> _time       equ 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
3183                                    <1> _mkdir      equ 14
3184                                    <1> _chmod      equ 15 ; TRDOS 386, Change Attributes (30/12/2017)
3185                                    <1> _rmdir      equ 16 ; TRDOS 386, Remove Directory (29/12/2017)
3186                                    <1> _break      equ 17
3187                                    <1> _drive      equ 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
3188                                    <1> _seek equ 19
3189                                    <1> _tell       equ 20
3190                                    <1> _mem  equ 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
3191                                    <1> _prompt     equ 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
3192                                    <1> _path equ 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
3193                                    <1> _env  equ 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
3194                                    <1> _stime      equ 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
3195                                    <1> _quit equ 26
3196                                    <1> _intr equ 27
3197                                    <1> _dir  equ 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
3198                                    <1> _emt  equ 29
3199                                    <1> _ldrvt      equ 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
3200                                    <1> _video  equ 31 ; TRDOS 386 Video Functions (16/05/2016)
3201                                    <1> _audio      equ 32 ; TRDOS 386 Video Functions (16/05/2016)
3202                                    <1> _timer      equ 33 ; TRDOS 386 Timer Functions (18/05/2016)
3203                                    <1> _sleep      equ 34 ; Retro UNIX 8086 v1 feature only !
3204                                    <1> _msg  equ 35 ; Retro UNIX 386 v1 feature only !
3205                                    <1> _geterr     equ 36 ; Retro UNIX 386 v1 feature only !
3206                                    <1> _fpsave equ 37 ; TRDOS 386 FPU state option (28/02/2017)
3207                                    <1> _pri  equ 38 ; change priority - TRDOS 386 (20/05/2016)
3208                                    <1> _rele equ 39 ; TRDOS 386 (19/05/2016)
3209                                    <1> _fff  equ 40 ; Find First File - TRDOS 386 (15/10/2016)
3210                                    <1> _fnf  equ 41 ; Find Next File - TRDOS 386 (15/10/2016)
3211                                    <1> _alloc      equ 42 ; Allocate memory - TRDOS 386 (19/02/2017)
3212                                    <1>              ; TRDOS 386 (19/02/2017) DMA buff fuctions
3213                                    <1> _dalloc equ 43 ; Deallocate mem - TRDOS 386 (19/02/2017)
3214                                    <1> _calbac equ 44 ; Set IRQ callback - TRDOS 386 (20/02/2017)
3215                                    <1> _dma  equ 45 ; DMA service - TRDOS 386 (20/08/2017)
3216                                    <1>
3217                                    <1> %macro sys 1-4
3218                                    <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3219                                    <1>     ; 03/09/2015
3220                                    <1>     ; 13/04/2015
3221                                    <1>     ; Retro UNIX 386 v1 system call.
3222                                    <1>     %if %0 >= 2
3223                                    <1>         mov ebx, %2
3224                                    <1>         %if %0 >= 3
3225                                    <1>             mov ecx, %3
3226                                    <1>             %if %0 = 4
3227                                    <1>                mov edx, %4
3228                                    <1>             %endif
3229                                    <1>         %endif
3230                                    <1>     %endif
3231                                    <1>     mov eax, %1
3232                                    <1>     ;int 30h
3233                                    <1>     int 40h ; TRDOS 386 (TRDOS v2.0)
3234                                    <1> %endmacro
3235                                    <1>
3236                                    <1> ; TRDOS 386 system calls, interrupt number
3237                                    <1> ; 25/12/2016
3238                                    <1> SYSCALL_INT_NUM   equ '40' ; '40h'
3239                                    <1>
3240                                    <1> ; 13/05/2015 - ERROR CODES
3241                                    <1> ERR_FILE_NOT_OPEN  equ 10 ; 'file not open !' error
3242                                    <1> ERR_FILE_ACCESS    equ 11 ; 'permission denied !' error
3243                                    <1> ; 14/05/2015
3244                                    <1> ERR_DIR_ACCESS     equ 11 ; 'permission denied !' error
3245                                    <1> ERR_FILE_NOT_FOUND equ 12 ; 'file not found !' error
3246                                    <1> ERR_TOO_MANY_FILES equ 13 ; 'too many open files !' error
3247                                    <1> ERR_DIR_EXISTS     equ 14 ; 'directory already exists !' error
3248                                    <1> ; 16/05/2015
3249                                    <1> ERR_DRV_NOT_RDY    equ 15 ; 'drive not ready !' error
3250                                    <1> ; 18/05/2015
3251                                    <1> ERR_DEV_NOT_RDY    equ 15 ; 'device not ready !' error
```

```
3252                            <1> ERR_DEV_ACCESS     equ 11 ; 'permission denied !' error
3253                            <1> ERR_DEV_NOT_OPEN   equ 10 ; 'device not open !' error
3254                            <1> ; 07/06/2015
3255                            <1> ERR_FILE_EOF    equ 16 ; 'end of file !' error
3256                            <1> ERR_DEV_VOL_SIZE   equ 16 ; 'out of volume !' error
3257                            <1> ; 09/06/2015
3258                            <1> ERR_DRV_READ    equ 17 ; 'disk read error !'
3259                            <1> ERR_DRV_WRITE         equ 18 ; 'disk write error !'
3260                            <1> ; 16/06/2015
3261                            <1> ERR_NOT_DIR     equ 19 ; 'not a (valid) directory !' error
3262                            <1> ERR_FILE_SIZE         equ 20 ; 'file size error !'
3263                            <1> ; 22/06/2015
3264                            <1> ERR_NOT_SUPERUSER  equ 11 ; 'permission denied !' error
3265                            <1> ERR_NOT_OWNER      equ 11 ; 'permission denied !' error
3266                            <1> ERR_NOT_FILE       equ 11 ; 'permission denied !' error
3267                            <1> ; 23/06/2015
3268                            <1> ERR_FILE_EXISTS    equ 14 ; 'file already exists !' error
3269                            <1> ERR_DRV_NOT_SAME   equ 21 ; 'not same drive !' error
3270                            <1> ERR_DIR_NOT_FOUND  equ 12 ; 'directory not found !' error
3271                            <1> ERR_NOT_EXECUTABLE equ 22 ; 'not executable file !' error
3272                            <1> ; 27/06/2015
3273                            <1> ERR_INV_PARAMETER  equ 23 ; 'invalid parameter !' error
3274                            <1> ERR_INV_DEV_NAME   equ 24 ; 'invalid device name !' error
3275                            <1> ; 29/06/2015
3276                            <1> ERR_TIME_OUT    equ 25 ; 'time out !' error
3277                            <1> ERR_DEV_NOT_RESP   equ 25 ; 'device not responding !' error
3278                            <1> ; 10/10/2016
3279                            <1> ERR_INV_FILE_NAME  equ 26 ; 'invalid file name !' error
3280                            <1> ERR_INV_FLAGS         equ 23 ; 'invalid flags !' error
3281                            <1> ; For code compatibility with previous version of TRDOS (2011)
3282                            <1> ; (Temporary error codes for current TRDOS 386 -2016- version)
3283                            <1> ERR_NO_MORE_FILES  equ 12 ; 'no more files !' error
3284                            <1> ERR_PATH_NOT_FOUND equ  3 ; 'path not found !' error
3285                            <1>                         ; 'dir not found !' ; TRDOS 8086
3286                            <1> ERR_NOT_FOUND:         equ  2 ; 'file not found !' ; TRDOS 8086
3287                            <1> ERR_DISK_SPACE         equ 39 ; 'out of volume !' TRDOS 8086
3288                            <1>                        ; 'insufficient disk space !' ; 27h
3289                            <1> ERR_DISK_WRITE         equ 30 ; 'disk write protected !' ; 16/10/2016
3290                            <1> ERR_ACCESS_DENIED  equ  5 ; 'access denied !' ; TRDOS 8086
3291                            <1> ; 28/02/2017
3292                            <1> ERR_PERM_DENIED        equ 11 ; 'permission denied !' error
3293                            <1> ; 18/05/2016
3294                            <1> ERR_MISC        equ 27 ; miscellaneous/other errors
3295                            <1> ; 15/10/2016
3296                            <1> ; TRDOS 8086 -> TRDOS 386 (0Bh -> 28)
3297                            <1> ERR_INV_FORMAT         equ 28 ; 'invalid format !' error
3298                            <1> ; TRDOS 8086 -> TRDOS 386 (0Dh -> 29)
3299                            <1> ERR_INV_DATA    equ 29 ; 'invalid data !' error
3300                            <1> ; TRDOS 8086 -> TRDOS 386 (0Eh -> 20)
3301                            <1> ERR_ZERO_LENGTH        equ 20  ; 'zero length !' error
3302                            <1> ; TRDOS 8086 -> TRDOS 386 (15h -> 17, 1Dh -> 18, 1Eh -> 17)
3303                            <1> ERR_DRV_NR_READ        equ 17 ; 'drive not ready or read error !'
3304                            <1> ERR_DRV_NR_WRITE   equ 18 ; 'drive not ready or write error !'
3305                            <1> ; 15/10/2016
3306                            <1> ERR_INV_PATH_NAME  equ 19 ; 'bad path name !' error
3307                            <1> ERR_BAD_CMD_ARG        equ  1 ; 'bad command argument !' ; TRDOS 8086
3308                            <1> ERR_INV_FNUMBER        equ  1 ; 'invalid function number !' ; TRDOS 8086
3309                            <1> ERR_BIG_FILE    equ  8 ; 'big file & out of memory ! ; TRDOS 8086
3310                            <1> ERR_BIG_DATA    equ  8 ; 'big data & out of memory ! ; TRDOS 8086
3311                            <1> ERR_CLUSTER     equ 35 ; 'cluster not available !' ; TRDOS 8086
3312                            <1> ERR_OUT_OF_MEMORY  equ  4 ; 'out of memory !'
3313                            <1>                         ; 'insufficient memory !'
3314                            <1> ERR_P_VIOLATION        equ 6 ; 'protection violation !'
3315                            <1> ERR_PAGE_FAULT         equ 224 ;'page fault !' ;0E0h
3316                            <1> ERR_SWP_DISK_READ            equ 40
3317                            <1> ERR_SWP_DISK_NOT_PRESENT   equ 41
3318                            <1> ERR_SWP_SECTOR_NOT_PRESENT equ 42
3319                            <1> ERR_SWP_NO_FREE_SPACE      equ 43
3320                            <1> ERR_SWP_DISK_WRITE         equ 44
3321                            <1> ERR_SWP_NO_PAGE_TO_SWAP    equ 45
3322                            <1> ; 10/04/2017
3323                            <1> ERR_BUFFER      equ 46  ; 'buffer error !'
3324                            <1> ; 28/08/2017 (20/08/2017)
3325                            <1> ERR_DMA               equ -1  ; DMA buffer (allocation/misc.) error!
3326                            <1>
3327                            <1> ; 26/08/2015
3328                            <1> ; 24/07/2015
3329                            <1> ; 24/06/2015
3330                            <1> MAX_ARG_LEN     equ 256 ; max. length of sys exec arguments
3331                            <1> ; 01/07/2015
3332                            <1> MAX_MSG_LEN     equ 255 ; max. msg length for 'sysmsg'
3333                            <1> ;
3334                            <1> ; 06/10/2016
3335                            <1> OPENFILES       equ 10  ; max. number of open files (system)
3336                            <1> ; 07/10/2016
3337                            <1> ;NUMOFDEVICES         equ 20  ; max. num of available devices (sys)
3338                            <1>
3083                                %include 'trdosk0.s' ; 04/01/2016
3084                            <1> ; ****************************************************************************
3085                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DEFINITIONS : trdosk0.s
3086                            <1> ; ----------------------------------------------------------------------------
3087                            <1> ; Last Update: 29/02/2016
3088                            <1> ; ----------------------------------------------------------------------------
3089                            <1> ; Beginning: 04/01/2016
3090                            <1> ; ----------------------------------------------------------------------------
3091                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3092                            <1> ; ----------------------------------------------------------------------------
3093                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3094                            <1> ; TRDOS2.ASM (09/11/2011)
3095                            <1> ; ****************************************************************************
3096                            <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
3097                            <1> ;
3098                            <1> ; Masterboot / Partition Table at Beginning+1BEh
3099                            <1> ptBootable       equ 0
3100                            <1> ptBeginHead      equ 1
```

```
3101                            <1> ptBeginSector    equ 2
3102                            <1> ptBeginCylinder  equ 3
3103                            <1> ptFileSystemID   equ 4
3104                            <1> ptEndHead        equ 5
3105                            <1> ptEndSector      equ 6
3106                            <1> ptEndCylinder    equ 7
3107                            <1> ptStartSector    equ 8
3108                            <1> ptSectors        equ 12
3109                            <1>
3110                            <1> ; Boot Sector Parameters at 7C00h
3111                            <1> DataArea1     equ -4
3112                            <1> DataArea2     equ -2
3113                            <1> BootStart     equ 0h
3114                            <1> OemName       equ 03h
3115                            <1> BytesPerSec   equ 0Bh
3116                            <1> SecPerClust   equ 0Dh
3117                            <1> ResSectors    equ 0Eh
3118                            <1> FATs          equ 10h
3119                            <1> RootDirEnts   equ 11h
3120                            <1> Sectors       equ 13h
3121                            <1> Media         equ 15h
3122                            <1> FATSecs       equ 16h
3123                            <1> SecPerTrack   equ 18h
3124                            <1> Heads         equ 1Ah
3125                            <1> Hidden1       equ 1Ch
3126                            <1> Hidden2       equ 1Eh
3127                            <1> HugeSec1      equ 20h
3128                            <1> HugeSec2      equ 22h
3129                            <1> DriveNumber   equ 24h
3130                            <1> Reserved1     equ 25h
3131                            <1> bootsignature equ 26h
3132                            <1> VolumeID      equ 27h
3133                            <1> VolumeLabel   equ 2Bh
3134                            <1> FileSysType   equ 36h
3135                            <1> Reserved2     equ 3Eh                           ; Starting cluster of P2000
3136                            <1>
3137                            <1> ; FAT32 BPB Structure
3138                            <1> FAT32_FAT_Size equ 36
3139                            <1> FAT32_RootFClust equ 44
3140                            <1> FAT32_FSInfoSec equ 48
3141                            <1> FAT32_DrvNum equ 64
3142                            <1> FAT32_BootSig equ 66
3143                            <1> FAT32_VolID equ 67
3144                            <1> FAT32_VolLab equ 71
3145                            <1> FAT32_FilSysType equ 82
3146                            <1>
3147                            <1> ; BIOS Disk Parameters
3148                            <1> DPDiskNumber  equ 0h
3149                            <1> DPDType       equ 1h
3150                            <1> DPReturn      equ 2h
3151                            <1> DPHeads       equ 3h
3152                            <1> DPCylinders   equ 4h
3153                            <1> DPSecPerTrack equ 6h
3154                            <1> DPDisks       equ 7h
3155                            <1> DPTableOff    equ 8h
3156                            <1> DPTableSeg    equ 0Ah
3157                            <1> DPNumOfSecs   equ 0Ch
3158                            <1>
3159                            <1> ; BIOS INT 13h Extensions (LBA extensions)
3160                            <1> ; Just After DP Data (DPDiskNumber+)
3161                            <1> DAP_PacketSize equ 10h  ; If extensions present, this byte will be >=10h
3162                            <1> DAP_Reserved1 equ 11h   ; Reserved Byte
3163                            <1> DAP_NumOfBlocks equ 12h ; Value of this byte must be 0 to 127
3164                            <1> DAP_Reserved2 equ 13h   ; Reserved Byte
3165                            <1> DAP_Destination equ 14h ; Address of Transfer Buffer as SEGMENT:OFFSET
3166                            <1> DAP_LBA_Address equ 18h ; LBA=(C1*H0+H1)*S0+S1-1
3167                            <1>                         ; C1= Selected Cylinder Number
3168                            <1>                         ; H0= Number Of Heads (Maximum Head Number + 1)
3169                            <1>                         ; H1= Selected Head Number
3170                            <1>                         ; S0= Maximum Sector Number
3171                            <1>                         ; S1= Selected Sector Number
3172                            <1>                         ; QUAD WORD
3173                            <1> ; DAP_Flat_Destination equ 20h ; 64 bit address, if value in 4h is FFFF:FFFFh
3174                            <1>                               ; QUAD WORD (Also, value in 0h must be 18h)
3175                            <1>                               ; TR-DOS will not use 64 bit Flat Address
3176                            <1>
3177                            <1> ; INT 13h Function 48h "Get Enhanced Disk Drive Parameters"
3178                            <1> ; Just After DP Data (DPDiskNumber+)
3179                            <1> GetDParams_48h equ 20h ; Word. Data Length, must be 26 (1Ah) for short data.
3180                            <1> GDP_48h_InfoFlag equ 22h ; Word
3181                            <1> ; Bit 1 = 1 -> The geometry returned in bytes 4-15 is valid.
3182                            <1> GDP_48h_NumOfPCyls equ 24h ; Double Word. Number physical cylinders.
3183                            <1> GDP_48h_NumOfPHeads equ 28h ; Double Word. Number of physical heads.
3184                            <1> GDP_48h_NumOfPSpT equ 2Ch ; Double word. Num of physical sectors per track.
3185                            <1> GDP_48h_LBA_Sectors equ 30h ; 8 bytes. Number of physical/LBA sectors.
3186                            <1> GDP_48h_BytesPerSec equ 38h ; Word. Number of bytes in a sector.
3187                            <1>
3188                            <1> ; TR-DOS Standalone Program Extensions to the DiskParams Block
3189                            <1> ; Just After DP Data (DPDiskNumber+)
3190                            <1> TRDP_CurrentSector equ 3Ah  ; DX:AX (LBA)
3191                            <1> TRDP_SectorCount equ 3Eh    ; CX (or Counter)
3192                            <1>
3193                            <1>
3194                            <1> ; DOS Logical Disks
3195                            <1> LD_Name equ 0
3196                            <1> LD_DiskType equ 1
3197                            <1> LD_PhyDrvNo equ 2
3198                            <1> LD_FATType equ 3
3199                            <1> LD_FSType equ 4
3200                            <1> LD_LBAYes equ 5
3201                            <1> LD_BPB equ 6
3202                            <1> LD_FATBegin equ 96
3203                            <1> LD_ROOTBegin equ 100
3204                            <1> LD_DATABegin equ 104
3205                            <1> LD_StartSector equ 108
```

```
3206                            <1> LD_TotalSectors equ 112
3207                            <1> LD_FreeSectors equ 116
3208                            <1> LD_Clusters equ 120
3209                            <1> LD_PartitionEntry equ 124
3210                            <1> LD_DParamEntry equ 125
3211                            <1> LD_MediaChanged equ 126
3212                            <1> LD_CDirLevel equ 127
3213                            <1> LD_CurrentDirectory equ 128
3214                            <1>
3215                            <1> ; Singlix FS Extensions to DOS Logical Disks
3216                            <1> ; 03/01/2010 (LD_BPB compatibility for CHS r/w)
3217                            <1>
3218                            <1> LD_FS_Name equ 0
3219                            <1> LD_FS_DiskType equ 1
3220                            <1> LD_FS_PhyDrvNo equ 2
3221                            <1> LD_FS_FATType equ 3
3222                            <1> LD_FS_FSType equ 4
3223                            <1> LD_FS_LBAYes equ 5
3224                            <1> LD_FS_BPB equ 6
3225                            <1> LD_FS_MediaAttrib equ 6
3226                            <1> LD_FS_VersionMajor equ 7
3227                            <1> LD_FS_RootDirD equ 8
3228                            <1> LD_FS_MATLocation equ 12
3229                            <1> LD_FS_Reserved1 equ 16 ;1 reserved byte
3230                            <1> LD_FS_BytesPerSec equ 17 ; LD_BPB + 0Bh
3231                            <1> LD_FS_Reserved2 equ 19 ;2 reserved byte
3232                            <1> LD_FS_DATLocation equ 20
3233                            <1> LD_FS_DATSectors equ 24
3234                            <1> LD_FS_Reserved3 equ 28 ;3 reserved word
3235                            <1> LD_FS_SecPerTrack equ 30 ; LD_BPB + 18h
3236                            <1> LD_FS_NumHeads equ 32    ; LD_BPB + 1Ah
3237                            <1> LD_FS_UnDelDirD equ 34
3238                            <1> LD_FS_Reserved4 equ 38 ;4 reserved word
3239                            <1> LD_FS_VolumeSerial equ 40
3240                            <1> LD_FS_VolumeName equ 44
3241                            <1> LD_FS_BeginSector equ 108
3242                            <1> LD_FS_VolumeSize equ 112
3243                            <1> LD_FS_FreeSectors equ 116
3244                            <1> LD_FS_FirstFreeSector equ 120
3245                            <1> LD_FS_PartitionEntry equ 124
3246                            <1> LD_FS_DParamEntry equ 125
3247                            <1> LD_FS_MediaChanged equ 126
3248                            <1> LD_FS_CDirLevel equ 127
3249                            <1> LD_FS_CDIR_Converted equ 128
3250                            <1>
3251                            <1> ; Valid FAT Types
3252                            <1> FS_FAT12 equ 1
3253                            <1> FS_FAT16_CHS equ 2
3254                            <1> FS_FAT32_CHS equ 3
3255                            <1> FS_FAT16_LBA equ 4
3256                            <1> FS_FAT32_LBA equ 5
3257                            <1>
3258                            <1> ; Cursor Location
3259                            <1> CCCpointer equ  0450h   ; BIOS data, current cursor column
3260                            <1> ; FAT Clusters EOC sign
3261                            <1> FAT12EOC equ 0FFFh
3262                            <1> FAT16EOC equ 0FFFFh
3263                            <1> ;FAT32EOC equ 0FFFFFFFh ; It is not direct usable for 8086 code
3264                            <1> ; BAD Cluster
3265                            <1> FAT12BADC equ 0FF7h
3266                            <1> FAT16BADC equ 0FFF7h
3267                            <1> ;FAT32BADC equ 0FFFFFF7h ; It is not direct usable for 8086 code
3268                            <1> ; MS-DOS FAT16 FS (Maximum Possible) Last Cluster Number= 0FFF6h
3269                            <1>
3270                            <1> ; TRFS
3271                            <1>
3272                            <1> bs_FS_JmpBoot equ 0 ; jmp short bsBootCode
3273                            <1>                     ; db 0EBh, db 3Fh, db 90h
3274                            <1> bs_FS_Identifier equ 3  ; db 'FS', db 0
3275                            <1> bs_FS_BytesPerSec equ 6 ; dw 512
3276                            <1> bs_FS_MediaAttrib equ 8 ; db 3
3277                            <1> bs_FS_PartitionID equ 9 ; db 0A1h
3278                            <1> bs_FS_VersionMaj equ 10 ; db 01h
3279                            <1> bs_FS_VersionMin equ 11 ; db 0
3280                            <1> bs_FS_BeginSector equ 12   ; dd 0
3281                            <1> bs_FS_VolumeSize equ 16 ; dd 2880
3282                            <1> bs_FS_StartupFD equ 20 ; dd 0
3283                            <1> bs_FS_MATLocation equ 24 ; dd 1
3284                            <1> bs_FS_RootDirD equ 28 ; dd 8
3285                            <1> bs_FS_SystemConfFD equ 32 ; dd 0
3286                            <1> bs_FS_SwapFD equ 36 ; dd 0
3287                            <1> bs_FS_UnDelDirD equ 40 ; dd 0
3288                            <1> bs_FS_DriveNumber equ 44 ; db 0
3289                            <1> bs_FS_LBA_Ready equ 45 ; db 0
3290                            <1> bs_FS_MagicWord equ 46
3291                            <1> bs_FS_SecPerTrack equ 46 ; db 0A1h
3292                            <1> bs_FS_Heads equ 47 ; db 01h
3293                            <1> bs_FS_OperationSys equ 48 ; db "TR-SINGLIX v1.0b"
3294                            <1> bs_FS_Terminator equ 64 ; db 0
3295                            <1> bs_FS_BootCode equ 65
3296                            <1>
3297                            <1> FS_MAT_DATLocation equ 12
3298                            <1> FS_MAT_DATScount equ 16
3299                            <1> FS_MAT_FreeSectors equ 20
3300                            <1> FS_MAT_FirstFreeSector equ 24
3301                            <1> FS_RDT_VolumeSerialNo equ 28
3302                            <1> FS_RDT_VolumeName equ 64
3303                            <1>
3304                            <1> ; FAT12 + FAT16 + FAT32
3305                            <1> BS_JmpBoot equ 0
3306                            <1> BS_OEMName equ 3
3307                            <1> BPB_BytsPerSec equ 11
3308                            <1> BPB_SecPerClust equ 13
3309                            <1> BPB_RsvdSecCnt equ 14
3310                            <1> BPB_NumFATs equ 16
```

```
                        <1> BPB_RootEntCnt equ 17
                        <1> BPB_TotalSec16 equ 19
                        <1> BPB_Media equ 21
                        <1> BPB_FATSz16 equ 22
                        <1> BPB_SecPerTrk equ 24
                        <1> BPB_NumHeads equ 26
                        <1> BPB_HiddSec equ 28
                        <1> BPB_TotalSec32 equ 32
                        <1>
                        <1> ; FAT12 and FAT16 only
                        <1> BS_DrvNum equ 36
                        <1> BS_Reserved1 equ 37
                        <1> BS_BootSig equ 38
                        <1> BS_VolID equ 39
                        <1> BS_VolLab equ 43
                        <1> BS_FilSysType equ 54 ; 8 bytes
                        <1> BS_BootCode equ 62
                        <1>
                        <1> ; FAT32 only
                        <1> BPB_FATSz32 equ 36 ; FAT32, 4 bytes
                        <1> BPB_ExtFlags equ 40 ; FAT32, 2 bytes
                        <1> BPB_FSVer equ 42 ; FAT32, 2 bytes
                        <1> BPB_RootClus equ 44 ; FAT32, 4 bytes
                        <1> BPB_FSInfo equ 48 ; FAT 32, 2 bytes
                        <1> BPB_BkBootSec equ 50 ; FAT32, 2 bytes
                        <1> BPB_Reserved equ 52 ; FAT32, 12 bytes
                        <1> BS_FAT32_DrvNum equ 64 ; FAT32, 1 byte
                        <1> BS_FAT32_Reserved1 equ 65 ; FAT32, 1 byte
                        <1> BS_FAT32_BootSig equ 66 ; FAT32, 1 byte
                        <1> BS_FAT32_VolID equ 67 ; FAT32, 4 bytes
                        <1> BS_FAT32_VolLab equ 71 ; FAT32, 11 bytes
                        <1> BS_FAT32_FilSysType equ 82 ; FAT32, 8 bytes
                        <1> BS_FAT32_BootCode equ 90
                        <1>
                        <1> ; 29/02/2016
                        <1> ;(FAT32 Free Cluster Count & First Free Cluster values)
                        <1> ;[BPB_Reserved] = Free Cluster Count (offset 52)
                        <1> ;[BPB_Reserved+4] = First Free Cluster (offset 56)
                        <1>
                        <1> BS_Validation equ 510
                        <1>
                        <1> ; 15/02/2016
                        <1> ; FILE.ASM - 09/10/2011
                        <1> ; Directory Entry Structure
                        <1> ; 29/10/2009 (According to Microsoft FAT32 File System Specification)
                        <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
    3084                    %include 'trdosk1.s' ; 04/01/2016
                        <1> ; ****************************************************************************
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.3) - SYS INIT : trdosk1.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 06/12/2020
                        <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>       ; 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
                        <1>
                        <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
00007306 B036           <1>       mov   al, 00110110b ; 36h
00007308 E643           <1>       out   43h, al
0000730A 31C0           <1>       xor   eax, eax  ; sub     al, al ; 0
0000730C E640           <1>       out   40h, al ; LB
0000730E E640           <1>       out   40h, al ; HB
                        <1>       ;
                        <1>       ; 30/03/2016
                        <1>       ; Clear Logical DOS Disk Description Tables Area
                        <1>       ;xor   eax, eax
00007310 BF00010900     <1>       mov   edi, Logical_DOSDisks
00007315 B980060000     <1>       mov   ecx, 6656/4 ; 26*256 = 6656 bytes
0000731A F3AB           <1>       rep   stosd ; 1664 times 4 bytes
                        <1>
0000731C B83F3A2F00     <1>       mov   eax, '?:/'
00007321 A3[B7890100]   <1>       mov   [Current_Dir_Drv], eax
                        <1>
                        <1>       ; Logical DRV INIT (only for hard disks)
00007326 E825040000     <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>
0000732B BE00010900          <1>       mov    esi, Logical_DOSDisks
00007330 B001                <1>       mov    al, 1 ; Initialization sign (invalid_fd_parameter)
00007332 83C67E              <1>       add    esi, LD_MediaChanged ; Media Change Status = 1 (init needed)
00007335 8806                <1>       mov    [esi], al ; A:
00007337 81C600010000        <1>       add    esi, 100h
0000733D 8806                <1>       mov    [esi], al ; B:
                            <1>
                            <1> _current_drive_bootdisk:
0000733F 8A15[CA6C0000]      <1>       mov    dl, [boot_drv] ; physical drive number
00007345 80FAFF              <1>       cmp    dl, 0FFh
00007348 740A                <1>       je     short _last_dos_diskno_check
                            <1> _boot_drive_check:
0000734A 80FA80              <1>       cmp    dl, 80h
0000734D 7218                <1>       jb     short _current_drive_a
0000734F 80EA7E              <1>       sub    dl, 7Eh ; C = 2 , D = 3
00007352 EB13                <1>       jmp    short _current_drive_a
                            <1>
                            <1> _last_dos_diskno_check:
00007354 8A15[823F0100]      <1>       mov    dl, [Last_DOS_DiskNo]
0000735A 80FA02              <1>       cmp    dl, 2
0000735D 7706                <1>       ja     short _current_drive_c
0000735F 7406                <1>       je     short _current_drive_a
00007361 30D2                <1>       xor    dl, dl ; A:
00007363 EB02                <1>       jmp    short _current_drive_a
                            <1>
                            <1> _current_drive_c:
00007365 B202                <1>       mov    dl, 2 ; C:
                            <1>
                            <1> _current_drive_a:
00007367 8815[CB6C0000]      <1>       mov    [drv], dl
0000736D BE[843F0100]        <1>         mov     esi, msg_CRLF_temp
00007372 E8AE000000          <1>       call   print_msg
                            <1>
00007377 8A15[CB6C0000]      <1>       mov    dl, [drv]
                            <1> _default_drive_c:
0000737D E82C0C0000          <1>       call   change_current_drive
00007382 731C                <1>       jnc    short _start_mainprog
                            <1>
                            <1> _drv_not_ready_error:
00007384 BE[3F420100]        <1>       mov    esi, msgl_drv_not_ready
00007389 E897000000          <1>       call   print_msg
                            <1>         ;jmp _end_of_mainprog
                            <1>
                            <1>       ; 20/01/2018
0000738E B202                <1>       mov    dl, 2
00007390 3815[CB6C0000]      <1>       cmp    [drv], dl
00007396 736B                <1>       jnb    short _end_of_mainprog
00007398 8815[CB6C0000]      <1>       mov    [drv], dl
0000739E EBDD                <1>       jmp    short _default_drive_c
                            <1>
                            <1> _start_mainprog:
                            <1>       ; 07/01/2017
                            <1>       ; 07/05/2016
                            <1>       ; 02/05/2016
                            <1>       ; 24/04/2016
                            <1>       ; Retro UNIX 386 v1, 'sys_init' (u0.s)
                            <1>       ; 23/06/2015
                            <1>
                            <1>       ; 02/05/2016
                            <1>       ; 24/04/2016
000073A0 66B80100            <1>       mov    ax, 1
000073A4 A2[B3030300]        <1>       mov    [u.uno], al
000073A9 66A3[4E030300]      <1>       mov    [mpid], ax
000073AF 66A3[20000300]      <1>       mov    [p.pid], ax
000073B5 A2[B0000300]        <1>       mov    [p.stat], al
000073BA C605[A8030300]04    <1>       mov    byte [u.quant], time_count  ; 07/01/2017
                            <1>       ;
000073C1 A1[F0880100]        <1>       mov    eax, [k_page_dir]
000073C6 A3[B8030300]        <1>       mov    [u.pgdir], eax ; reset
                            <1>       ;
000073CB E8D7E6FFFF          <1>       call   allocate_page
000073D0 0F82B5000000        <1>       jc     panic
000073D6 A3[B4030300]        <1>       mov    [u.upage], eax ; user structure page
000073DB A3[C0000300]        <1>       mov    [p.upage], eax
000073E0 E83CE7FFFF          <1>       call   clear_page
                            <1>       ;
                            <1>       ; 24/08/2015
000073E5 FE0D[5B030300]      <1>       dec    byte [sysflg] ; FFh = ready for system call
                            <1>                            ; 0 = executing a system call
                            <1>       ; 13/04/2016
                            <1>       ; Clear Environment Variables Page/Area
000073EB BF00300900          <1>       mov    edi, Env_Page ; 93000h
000073F0 B980000000          <1>       mov    ecx, Env_Page_Size / 4    ; 512/4  (4096/4)

000073F5 31C0                <1>       xor    eax, eax
000073F7 F3AB                <1>       rep    stosd
                            <1>
                            <1>       ; 14/04/2016
000073F9 E807350000          <1>       call   mainprog_startup_configuration
                            <1>
000073FE E8EC0C0000          <1>         call    dos_prompt
                            <1>
                            <1> _end_of_mainprog:
00007403 BE[843F0100]        <1>         mov     esi, msg_CRLF_temp
00007408 E818000000          <1>       call   print_msg
0000740D BE[8A3F0100]        <1>       mov    esi, mainprog_Version
00007412 E80E000000          <1>       call   print_msg
                            <1>       ; 24/01/2016
00007417 28E4                <1>       sub    ah, ah
00007419 E8C59AFFFF          <1>       call   int16h ; call getch
```

```
3236 0000741E E9699FFFFF          <1>       jmp    cpu_reset
3237                              <1>
3238 00007423 EBFE                <1> infinitiveloop: jmp short infinitiveloop
3239                              <1>
3240                              <1> print_msg:
3241                              <1>       ; 13/05/2016
3242                              <1>       ; 04/01/2016
3243                              <1>       ; 01/07/2015
3244                              <1>       ; 13/03/2015 (Retro UNIX 386 v1)
3245                              <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
3246                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
3247                              <1>       ;
3248 00007425 8A3D[1E890100]      <1>       mov    bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
3249                              <1>       ;mov   bl, 07h ; Black background, light gray forecolor
3250                              <1>
3251 0000742B AC                  <1>       lodsb
3252                              <1> pmsg1:
3253 0000742C 56                  <1>       push   esi
3254                              <1>       ;mov   bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
3255 0000742D B307                <1>       mov    bl, 07h ; Black background, light gray forecolor
3256 0000742F E84EAEFFFF          <1>       call   _write_tty
3257 00007434 5E                  <1>       pop    esi
3258 00007435 AC                  <1>       lodsb
3259 00007436 20C0                <1>       and    al, al
3260 00007438 75F2                <1>       jnz    short pmsg1
3261 0000743A C3                  <1>       retn
3262                              <1>
3263                              <1> clear_screen:
3264                              <1>       ; 06/12/2020
3265                              <1>       ; 03/12/2020 (TRDOS 386 v2.0.3)
3266                              <1>       ; 13/05/2016
3267                              <1>       ; 30/01/2016
3268                              <1>       ; 24/01/2016
3269                              <1>       ; 04/01/2016
3270 0000743B 0FB61D[1E890100]    <1>       movzx  ebx, byte [ACTIVE_PAGE] ; video page number (0 to 7)
3271 00007442 8AA3[AB6E0000]      <1>       mov    ah, [ebx+vmode] ; default = 03h (80x25 text)
3272 00007448 80FC04              <1>       cmp    ah, 4
3273 0000744B 7205                <1>       jb     short cls1
3274 0000744D 80FC07              <1>       cmp    ah, 7
3275 00007450 7530                <1>       jne    short vga_clear
3276                              <1> cls1:
3277                              <1>       ;mov   bh, bl
3278                              <1>       ;mov   bl, 7
3279 00007452 3A25[9A6E0000]      <1>       cmp    ah, [CRT_MODE] ; current video mode ?
3280 00007458 740E                <1>       je     short cls2 ; yes (current video mode = 3)
3281                              <1>       ;;call set_mode_3 ; set video mode to 3 (& clear screen)
3282                              <1>       ;;retn
3283                              <1>       ; 06/12/2020
3284 0000745A 803D[1C120300]00    <1>       cmp    byte [pmi32], 0
3285 00007461 771F                <1>       ja     short vga_clear
3286 00007463 E924AEFFFF          <1>       jmp    set_mode_3
3287                              <1> cls2:
3288 00007468 88DF                <1>       mov    bh, bl ; video page (0 to 7)
3289 0000746A B307                <1>       mov    bl, 07h ; attribute to be used on blanked line
3290 0000746C 28C0                <1>       sub    al, al ; 0 =  entire window
3291 0000746E 6631C9              <1>       xor    cx, cx
3292 00007471 66BA4F18            <1>       mov    dx, 184Fh
3293 00007475 E859ABFFFF          <1>       call   _scroll_up ; 24/01/2016
3294                              <1>       ;
3295                              <1>       ;mov   bh, [ACTIVE_PAGE] ; video page number (0 to 7)
3296 0000747A 6631D2              <1>       xor    dx, dx
3297                              <1>       ;call  _set_cpos ; 24/01/2016
3298                              <1>       ;;retn
3299                              <1>       ; 03/12/2020
3300 0000747D E998AEFFFF          <1>       jmp    _set_cpos ; returns to the caller of this proc
3301                              <1> ;cls3:
3302                              <1> ;     retn
3303                              <1>       ; 06/12/2020
3304                              <1> vga_clear:
3305                              <1>       ; 03/12/2020
3306                              <1>       ; set mode by using _int10h
3307                              <1>       ; (also clears screen)
3308 00007482 88E0                <1>       mov    al, ah
3309 00007484 28E4                <1>       sub    ah, ah  ; set current video mode
3310                              <1>       ;call  _int10h ; simulates int 10h in TRDOS 386 kernel
3311                              <1>       ;jmp   short cls3
3312 00007486 E978A2FFFF          <1>       jmp    _int10h ; returns to the caller of this proc
3313                              <1>
3314                              <1> panic:
3315                              <1>       ; 13/05/2016 (TRDOS 386 = TRDOS v2)
3316                              <1>       ; 13/03/2015 (Retro UNIX 386 v1)
3317                              <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
3318 0000748B BE[474C0100]        <1>       mov    esi, panic_msg
3319 00007490 E890FFFFFF          <1>       call   print_msg
3320                              <1> key_to_reboot:
3321                              <1>         ; 24/01/2016
3322 00007495 28E4                <1>         sub     ah, ah
3323 00007497 E8479AFFFF          <1>         call    int16h ; call   getch
3324                              <1>         ; wait for a character from the current tty
3325                              <1>       ;
3326 0000749C B00A                <1>       mov    al, 0Ah
3327 0000749E 8A3D[1E890100]      <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
3328 000074A4 B307                <1>       mov    bl, 07h ; Black background,
3329                              <1>                      ; light gray forecolor
3330 000074A6 E8D7ADFFFF          <1>       call   _write_tty
3331 000074AB E9DC9EFFFF          <1>       jmp    cpu_reset
3332                              <1>
3333                              <1> ctrlbrk:
3334                              <1>       ; 12/11/2015
3335                              <1>       ; 13/03/2015 (Retro UNIX 386 v1)
3336                              <1>       ; 06/12/2013 (Retro UNIX 8086 v1)
3337                              <1>       ;
3338                              <1>       ; INT 1Bh (control+break) handler
3339                              <1>       ;
3340                              <1>             ; Retro Unix 8086 v1 feature only!
```

```
                                <1>             ;
000074B0 66833D[AA030300]00     <1>       cmp   word [u.intr], 0
000074B8 7645                   <1>       jna   short cbrk4
                                <1> cbrk0:
                                <1>       ; 12/11/2015
                                <1>       ; 06/12/2013
000074BA 66833D[AC030300]00     <1>       cmp   word [u.quit], 0
000074C2 743B                   <1>       jz    short cbrk4
                                <1>       ;
                                <1>       ; 20/09/2013
000074C4 6650                   <1>       push  ax
000074C6 A0[1E890100]           <1>       mov   al, [ptty]
                                <1>       ;
                                <1>       ; 12/11/2015
                                <1>       ;
                                <1>       ; ctrl+break (EOT, CTRL+D) from serial port
                                <1>       ; or ctrl+break from console (pseudo) tty
                                <1>       ; (!redirection!)
                                <1>       ;
000074CB 3C08                   <1>       cmp   al, 8 ; serial port tty nums > 7
000074CD 7211                   <1>         jb      short cbrk1 ; console (pseudo) tty
                                <1>       ;
                                <1>       ; Serial port interrupt handler sets [ptty]
                                <1>       ; to the port's tty number (as temporary).
                                <1>       ;
                                <1>       ; If active process is using a stdin or
                                <1>       ; stdout redirection (by the shell),
                                <1>         ; console tty keyboard must be available
                                <1>       ; to terminate running process,
                                <1>       ; in order to prevent a deadlock.
                                <1>       ;
000074CF 52                     <1>       push  edx
000074D0 0FB615[B3030300]       <1>       movzx edx, byte [u.uno]
000074D7 3A82[7F000300]         <1>       cmp     al, [edx+p.ttyc-1] ; console tty (rw)
000074DD 5A                     <1>       pop   edx
000074DE 7412                   <1>       je    short cbrk2
                                <1> cbrk1:
000074E0 FEC0                   <1>       inc   al  ; [u.ttyp] : 1 based tty number
                                <1>       ; 06/12/2013
000074E2 3A05[94030300]         <1>       cmp   al, [u.ttyp] ; recent open tty (r)
000074E8 7408                   <1>       je    short cbrk2
000074EA 3A05[95030300]         <1>         cmp     al, [u.ttyp+1] ; recent open tty (w)
000074F0 750B                   <1>       jne   short cbrk3
                                <1> cbrk2:
                                <1>       ;; 06/12/2013
                                <1>       ;mov  ax, [u.quit]
                                <1>       ;and  ax, ax
                                <1>       ;jz   short cbrk3
                                <1>       ;
000074F2 6631C0                 <1>       xor   ax, ax ; 0
000074F5 6648                   <1>       dec   ax
                                <1>       ; 0FFFFh = 'ctrl+brk' keystroke
000074F7 66A3[AC030300]         <1>       mov   [u.quit], ax
                                <1> cbrk3:
000074FD 6658                   <1>       pop   ax
                                <1> cbrk4:
000074FF C3                     <1>       retn
                                <1>
                                <1>
                                <1> ; 31/12/2017
                                <1> ; TRDOS 386 - 30/12/2017
                                <1> %define get_rtc_date RTC_40
                                <1> %define get_rtc_time RTC_20
                                <1> %define     set_rtc_date RTC_50
                                <1> %define set_rtc_time RTC_30
                                <1> get_rtc_date_time:
                                <1> ; Retro UNIX 8086 v1 - UNIX.ASM (01/09/2014)
                                <1> ;epoch:
                                <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit version)
                                <1>       ; 09/04/2013 (Retro UNIX 8086 v1 - UNIX.ASM)
                                <1>       ; 'epoch' procedure prototype:
                                <1>       ;                 UNIXCOPY.ASM, 10/03/2013
                                <1>       ; 14/11/2012
                                <1>       ; unixboot.asm (boot file configuration)
                                <1>       ; version of "epoch" procedure in "unixproc.asm"
                                <1>       ; 21/7/2012
                                <1>       ; 15/7/2012
                                <1>       ; 14/7/2012
                                <1>       ; Erdogan Tan - RETRO UNIX v0.1
                                <1>       ; compute current date and time as UNIX Epoch/Time
                                <1>       ; UNIX Epoch: seconds since 1/1/1970 00:00:00
                                <1>       ;
                                <1>         ;  ((Modified registers: EAX, EDX, ECX, EBX))
                                <1>       ;
                                <1>
00007500 E879F4FFFF             <1>       call  get_rtc_time        ; Return Current Time
00007505 86E9                   <1>         xchg        ch,cl
00007507 66890D[BE850100]       <1>         mov [hour], cx
0000750E 86F2                   <1>         xchg        dh,dl
00007510 668915[C2850100]       <1>         mov [second], dx
                                <1>       ;
00007517 E8D3F4FFFF             <1>         call        get_rtc_date        ; Return Current Date
0000751C 86E9                   <1>         xchg        ch,cl
0000751E 66890D[B8850100]       <1>         mov [year], cx
00007525 86F2                   <1>         xchg        dh,dl
00007527 668915[BA850100]       <1>         mov [month], dx
                                <1>       ;
0000752E 66B93030               <1>       mov   cx, 3030h
                                <1>       ;
00007532 A0[BE850100]           <1>       mov   al, [hour] ; Hour
                                <1>             ; AL <= BCD number)
00007537 D410                   <1>         db  0D4h,10h            ; Undocumented inst. AAM
                                <1>                                 ; AH = AL / 10h
                                <1>                                 ; AL = AL MOD 10h
```

```
3446 00007539 D50A              <1>         aad ; AX= AH*10+AL
3447 0000753B A2[BE850100]      <1>       mov   [hour], al
3448 00007540 A0[BF850100]      <1>       mov   al, [hour+1] ; Minute
3449                            <1>             ; AL <= BCD number)
3450 00007545 D410              <1>         db  0D4h,10h            ; Undocumented inst. AAM
3451                            <1>                                 ; AH = AL / 10h
3452                            <1>                                 ; AL = AL MOD 10h
3453 00007547 D50A              <1>         aad ; AX= AH*10+AL
3454 00007549 A2[C0850100]      <1>       mov   [minute], al
3455 0000754E A0[C2850100]      <1>       mov   al, [second] ; Second
3456                            <1>             ; AL <= BCD number)
3457 00007553 D410              <1>         db  0D4h,10h            ; Undocumented inst. AAM
3458                            <1>                                 ; AH = AL / 10h
3459                            <1>                                 ; AL = AL MOD 10h
3460 00007555 D50A              <1>         aad ; AX= AH*10+AL
3461 00007557 A2[C2850100]      <1>       mov   [second], al
3462 0000755C 66A1[B8850100]    <1>       mov   ax, [year] ; Year (century)
3463 00007562 6650              <1>         push      ax
3464                            <1>             ; AL <= BCD number)
3465 00007564 D410              <1>         db  0D4h,10h            ; Undocumented inst. AAM
3466                            <1>                                 ; AH = AL / 10h
3467                            <1>                                 ; AL = AL MOD 10h
3468 00007566 D50A              <1>         aad ; AX= AH*10+AL
3469 00007568 B464              <1>       mov   ah, 100
3470 0000756A F6E4              <1>       mul   ah
3471 0000756C 66A3[B8850100]    <1>       mov   [year], ax
3472 00007572 6658              <1>       pop   ax
3473 00007574 88E0              <1>       mov   al, ah
3474                            <1>             ; AL <= BCD number)
3475 00007576 D410              <1>         db  0D4h,10h            ; Undocumented inst. AAM
3476                            <1>                                 ; AH = AL / 10h
3477                            <1>                                 ; AL = AL MOD 10h
3478 00007578 D50A              <1>         aad ; AX= AH*10+AL
3479 0000757A 660105[B8850100]  <1>       add   [year], ax
3480 00007581 A0[BA850100]      <1>       mov   al, [month] ; Month
3481                            <1>             ; AL <= BCD number)
3482 00007586 D410              <1>         db  0D4h,10h            ; Undocumented inst. AAM
3483                            <1>                                 ; AH = AL / 10h
3484                            <1>                                 ; AL = AL MOD 10h
3485 00007588 D50A              <1>         aad ; AX= AH*10+AL
3486 0000758A A2[BA850100]      <1>       mov   [month], al
3487 0000758F A0[BB850100]      <1>         mov     al, [month+1]         ; Day
3488                            <1>             ; AL <= BCD number)
3489 00007594 D410              <1>         db  0D4h,10h            ; Undocumented inst. AAM
3490                            <1>                                 ; AH = AL / 10h
3491                            <1>                                 ; AL = AL MOD 10h
3492 00007596 D50A              <1>         aad ; AX= AH*10+AL
3493 00007598 A2[BC850100]      <1>         mov     [day], al
3494                            <1>
3495 0000759D C3                <1>       retn  ; 30/12/2017
3496                            <1>
3497                            <1> epoch:
3498 0000759E E85DFFFFFF        <1>       call  get_rtc_date_time ; TRDOS 386 - 30/12/2017
3499                            <1>
3500                            <1> convert_to_epoch:
3501                            <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
3502                            <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit modification)
3503                            <1>       ; 09/04/2013 (Retro UNIX 8086 v1)
3504                            <1>       ;
3505                            <1>       ; ((Modified registers: EAX, EDX, EBX))
3506                            <1>       ;
3507                            <1>       ; Derived from DALLAS Semiconductor
3508                            <1>       ; Application Note 31 (DS1602/DS1603)
3509                            <1>       ; 6 May 1998
3510 000075A3 29C0              <1>       sub   eax, eax
3511 000075A5 66A1[B8850100]    <1>       mov   ax, [year]
3512 000075AB 662DB207          <1>       sub   ax, 1970
3513 000075AF BA6D010000        <1>       mov   edx, 365
3514 000075B4 F7E2              <1>       mul   edx
3515 000075B6 31DB              <1>       xor   ebx, ebx
3516 000075B8 8A1D[BA850100]    <1>       mov   bl, [month]
3517 000075BE FECB              <1>       dec   bl
3518 000075C0 D0E3              <1>       shl   bl, 1
3519                            <1>       ;sub  edx, edx
3520 000075C2 668B93[C4850100]  <1>       mov   dx, [EBX+DMonth]
3521 000075C9 8A1D[BC850100]    <1>         mov     bl, [day]
3522 000075CF FECB              <1>       dec   bl
3523 000075D1 01D0              <1>       add   eax, edx
3524 000075D3 01D8              <1>       add   eax, ebx
3525                            <1>                     ; EAX = days since 1/1/1970
3526 000075D5 668B15[B8850100]  <1>       mov   dx, [year]
3527 000075DC 6681EAB107        <1>       sub   dx, 1969
3528 000075E1 66D1EA            <1>       shr   dx, 1
3529 000075E4 66D1EA            <1>       shr   dx, 1
3530                            <1>             ; (year-1969)/4
3531 000075E7 01D0              <1>       add   eax, edx
3532                            <1>                     ; + leap days since 1/1/1970
3533 000075E9 803D[BA850100]02  <1>       cmp   byte [month], 2     ; if past february
3534 000075F0 7610              <1>       jna   short cte1
3535 000075F2 668B15[B8850100]  <1>       mov   dx, [year]
3536 000075F9 6683E203          <1>       and   dx, 3 ; year mod 4
3537 000075FD 7503              <1>       jnz   short cte1
3538                            <1>                     ; and if leap year
3539 000075FF 83C001            <1>       add   eax, 1        ; add this year's leap day (february 29)
3540                            <1> cte1:                   ; compute seconds since 1/1/1970
3541 00007602 BA18000000        <1>       mov   edx, 24
3542 00007607 F7E2              <1>       mul   edx
3543 00007609 8A15[BE850100]    <1>       mov   dl, [hour]
3544 0000760F 01D0              <1>       add   eax, edx
3545                            <1>             ; EAX = hours since 1/1/1970 00:00:00
3546                            <1>       ;mov  ebx, 60
3547 00007611 B33C              <1>       mov   bl, 60
3548 00007613 F7E3              <1>       mul   ebx
3549 00007615 8A15[C0850100]    <1>       mov   dl, [minute]
3550 0000761B 01D0              <1>       add   eax, edx
```

```
                                <1>             ; EAX = minutes since 1/1/1970 00:00:00
                                <1>       ;mov   ebx, 60
0000761D F7E3                   <1>       mul    ebx
0000761F 8A15[C2850100]         <1>       mov    dl, [second]
00007625 01D0                   <1>       add    eax, edx
                                <1>             ; EAX -> seconds since 1/1/1970 00:00:00
00007627 C3                     <1>       retn
                                <1>
                                <1> ;set_date_time:
                                <1> convert_from_epoch:
                                <1>       ; 31/12/2017 (v2.0.0)
                                <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit version)
                                <1>       ; 20/06/2013 (Retro UNIX 8086 v1)
                                <1>       ; 'convert_from_epoch' procedure prototype:
                                <1>       ;                 UNIXCOPY.ASM, 10/03/2013
                                <1>       ;
                                <1>       ; ((Modified registers: EAX, EDX, ECX, EBX))
                                <1>       ;
                                <1>       ; Derived from DALLAS Semiconductor
                                <1>       ; Application Note 31 (DS1602/DS1603)
                                <1>       ; 6 May 1998
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ; EAX = Unix (Epoch) Time
                                <1>       ;
00007628 31D2                   <1>       xor    edx, edx
0000762A B93C000000             <1>       mov    ecx, 60
0000762F F7F1                   <1>       div    ecx
                                <1>       ;mov   [imin], eax   ; whole minutes
                                <1>                            ; since 1/1/1970
00007631 668915[C2850100]       <1>       mov    [second], dx  ; leftover seconds
00007638 29D2                   <1>       sub    edx, edx
0000763A F7F1                   <1>       div    ecx
                                <1>       ;mov   [ihrs], eax   ; whole hours
                                <1>       ;                    ; since 1/1/1970
0000763C 668915[C0850100]       <1>       mov    [minute], dx  ; leftover minutes
00007643 31D2                   <1>       xor    edx, edx
                                <1>       ;mov   cx, 24
00007645 B118                   <1>       mov    cl, 24
00007647 F7F1                   <1>       div    ecx
                                <1>       ;mov   [iday], ax   ; whole days
                                <1>                           ; since 1/1/1970
00007649 668915[BE850100]       <1>       mov    [hour], dx   ; leftover hours
00007650 05DB020000             <1>       add    eax, 365+366 ; whole day since
                                <1>                           ; 1/1/1968
                                <1>       ;mov   [iday], ax
00007655 50                     <1>       push   eax
00007656 29D2                   <1>       sub    edx, edx
00007658 B9B5050000             <1>       mov    ecx, (4*365)+1 ; 4 years = 1461 days
0000765D F7F1                   <1>       div    ecx
0000765F 59                     <1>       pop    ecx
                                <1>       ;mov   [lday], ax   ; count of quadyrs (4 years)
00007660 6652                   <1>       push   dx
                                <1>       ;mov   [qday], dx   ; days since quadyr began
00007662 6683FA3C               <1>       cmp    dx, 31 + 29  ; if past feb 29 then
00007666 F5                     <1>       cmc                 ; add this quadyr's leap day
00007667 83D000                 <1>       adc    eax, 0       ; to # of qadyrs (leap days)
                                <1>       ;mov   [lday], ax   ; since 1968
                                <1>       ;mov   cx, [iday]
0000766A 91                     <1>       xchg   ecx, eax     ; ECX = lday, EAX = iday
0000766B 29C8                   <1>       sub    eax, ecx     ; iday - lday
0000766D B96D010000             <1>       mov    ecx, 365
00007672 31D2                   <1>       xor    edx, edx
                                <1>       ; EAX = iday-lday, EDX = 0
00007674 F7F1                   <1>       div    ecx
                                <1>       ;mov   [iyrs], ax   ; whole years since 1968
                                <1>       ;jday = iday - (iyrs*365) - lday
                                <1>       ;mov [jday], dx      ; days since 1/1 of current year
                                <1>       ;add   eax, 1968
00007676 6605B007               <1>       add    ax, 1968     ; compute year
0000767A 66A3[B8850100]         <1>       mov    [year], ax
00007680 6689D1                 <1>       mov    cx, dx
                                <1>       ;mov   dx, [qday]
00007683 665A                   <1>       pop    dx
00007685 6681FA6D01             <1>       cmp    dx, 365           ; if qday <= 365 and qday >= 60
0000768A 7709                   <1>       ja     short cfe1   ; jday = jday +1
0000768C 6683FA3C               <1>       cmp    dx, 60       ; if past 2/29 and leap year then
00007690 F5                     <1>         cmc              ; add a leap day to the # of whole
00007691 6683D100               <1>       adc    cx, 0        ; days since 1/1 of current year
                                <1> cfe1:
                                <1>       ;mov   [jday], cx
00007695 66BB0C00               <1>       mov    bx, 12       ; estimate month
00007699 66BA6E01               <1>       mov    dx, 366      ; mday, max. days since 1/1 is 365
0000769D 6683E003               <1>       and    ax, 11b      ; year mod 4 (and dx, 3)
                                <1> cfe2: ; Month calculation  ; 0 to 11  (11 to 0)
000076A1 6639D1                 <1>       cmp    cx, dx       ; mday = # of days passed from 1/1
000076A4 731D                   <1>       jnb    short cfe3
000076A6 664B                   <1>       dec    bx           ; month = month - 1
000076A8 66D1E3                 <1>       shl    bx, 1
000076AB 668B93[C4850100]       <1>       mov    dx, [EBX+DMonth] ; # elapsed days at 1st of month
000076B2 66D1EB                 <1>       shr    bx, 1        ; bx = month - 1 (0 to 11)
000076B5 6683FB01               <1>       cmp    bx, 1        ; if month > 2 and year mod 4  = 0
000076B9 76E6                   <1>       jna    short cfe2   ; then mday = mday + 1
000076BB 08C0                   <1>       or     al, al       ; if past 2/29 and leap year then
000076BD 75E2                   <1>       jnz    short cfe2   ; add leap day (to mday)
000076BF 6642                   <1>       inc    dx           ; mday = mday + 1
000076C1 EBDE                   <1>       jmp    short cfe2
                                <1> cfe3:
000076C3 6643                   <1>       inc    bx          ; -> bx = month, 1 to 12
000076C5 66891D[BA850100]       <1>       mov    [month], bx
000076CC 6629D1                 <1>       sub    cx, dx      ; day = jday - mday + 1
000076CF 6641                   <1>       inc    cx
000076D1 66890D[BC850100]       <1>       mov    [day], cx
                                <1>
```

```
                        <1>       ; eax, ebx, ecx, edx is changed at return
                        <1>       ; output ->
                        <1>       ; [year], [month], [day], [hour], [minute], [second]
                        <1>
000076D8 C3             <1>       retn   ; 31/12/2017 (TRDOS 386)
                        <1>
                        <1> set_rtc_date_time:
                        <1>       ; 31/12/2017 (v2.0.0)
                        <1>       ; 30/12/2017 (TRDOS 386)
                        <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit version)
                        <1>       ; 20/06/2013 (Retro UNIX 8086 v1)
000076D9 E80F000000     <1>       call   set_date_bcd
                        <1>       ; Set real-time clock date
000076DE E839F3FFFF     <1>       call   set_rtc_date ; RTC_50
                        <1>       ; Set real-time clock time
000076E3 E832000000     <1>       call   set_time_bcd
000076E8 E9C0F2FFFF     <1>       jmp    set_rtc_time ; RTC_30
                        <1>
                        <1> ; 31/12/2017
                        <1> set_date_bcd:
000076ED A0[B9850100]   <1>         mov     al, [year+1]
000076F2 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
000076F4 D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
000076F6 88C5           <1>       mov    ch, al ; century (BCD)
000076F8 A0[B8850100]   <1>       mov    al, [year]
000076FD D40A           <1>       aam    ; ah = al / 10, al = al mod 10
000076FF D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007701 88C1           <1>       mov    cl, al ; year (BCD)
00007703 A0[BA850100]   <1>         mov  al, [month]
00007708 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
0000770A D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
0000770C 88C6           <1>       mov    dh, al ; month (BCD)
0000770E A0[BC850100]   <1>       mov    al, [day]
00007713 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
00007715 D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007717 88C6           <1>       mov    dh, al ; day (BCD)
00007719 C3             <1>       retn   ; 30/12/2017
                        <1>
                        <1> ; 31/12/2017
                        <1> set_time_bcd:
                        <1>         ; Read real-time clock time
                        <1>       ; (get day light saving time bit status)
0000771A FA             <1>       cli
0000771B E84EF4FFFF     <1>       CALL   UPD_IPR             ; CHECK FOR UPDATE IN PROCESS
                        <1>       ; cf = 1 -> al = 0
00007720 7207           <1>         jc      short stime1
00007722 B00B           <1>       MOV    AL,CMOS_REG_B       ; ADDRESS ALARM REGISTER
00007724 E860F4FFFF     <1>       CALL   CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
                        <1> stime1:
00007729 FB             <1>       sti
0000772A 2401           <1>       AND    AL,00000001B        ; MASK FOR VALID DSE BIT
0000772C 88C2           <1>       MOV    DL,AL               ; SET [DL] TO ZERO FOR NO DSE BIT
                        <1>       ; DL = 1 or 0 (day light saving time)
                        <1>       ;
0000772E A0[BE850100]   <1>       mov    al, [hour]
00007733 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
00007735 D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007737 88C5           <1>       mov    ch, al ; hour (BCD)
00007739 A0[C0850100]   <1>         mov     al, [minute]
0000773E D40A           <1>       aam    ; ah = al / 10, al = al mod 10
00007740 D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
00007742 88C1           <1>       mov    cl, al       ; minute (BCD)
00007744 A0[C2850100]   <1>         mov     al, [second]
00007749 D40A           <1>       aam    ; ah = al / 10, al = al mod 10
0000774B D510           <1>       db     0D5h,10h     ; Undocumented inst. AAD
                        <1>                           ; AL = AH * 10h + AL
0000774D 88C6           <1>       mov    dh, al      ; second (BCD)
0000774F C3             <1>       retn   ; 30/12/2017
                            %include 'trdosk2.s' ; 04/01/2016
                        <1> ; ****************************************************************************
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.2) - DRV INIT : trdosk2.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 30/08/2020
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Beginning: 04/01/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.14 (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 (c) 2009-2011 Erdogan TAN  [26/09/2009] Last Update: 07/08/2011
                        <1> ;
                        <1>
                        <1> ldrv_init: ; Logical Drive Initialization
                        <1>       ; 30/08/2020
                        <1>       ; 25/08/2020
                        <1>       ; 11/08/2020, 13/08/2020
                        <1>       ; 17/07/2020, 20/07/2020
                        <1>       ; 14/07/2020, 15/07/2020
                        <1>       ; 30/01/2018
                        <1>       ; 27/12/2017
                        <1>       ; 12/02/2016
                        <1>       ; 06/01/2016
                        <1>       ;     ('diskinit.inc', 'diskio.inc' integration)
                        <1>       ; 04/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 07/08/2011
                        <1>       ; 20/09/2009
                        <1>       ; 2005
```

```
3116                                    <1>
3117                                    <1>     ; 15/07/2020
3118                                    <1>     ;movzx ecx, byte [HF_NUM] ; number of fixed disks
3119                                    <1>     ;cmp   cl, 1
3120                                    <1>     ;jnb   short load_hd_partition_tables
3121                                    <1>
3122 00007750 A0[8C890100]              <1>     mov    al, [HF_NUM] ; number of fixed disks
3123 00007755 20C0                      <1>     and    al, al
3124 00007757 7501                      <1>     jnz    short load_hd_partition_tables
3125                                    <1>
3126                                    <1>     ; no any hard disks
3127 00007759 C3                        <1>     retn
3128                                    <1>
3129                                    <1> load_hd_partition_tables:
3130                                    <1>     ;mov   esi, [HDPM_TBL_VEC] ; primary master disk FDPT
3131                                    <1>     ; 15/07/2020
3132 0000775A BE[90890100]              <1>     mov    esi, HDPM_TBL_VEC
3133 0000775F BF[B68D0100]              <1>     mov    edi, PTable_hd0
3134 00007764 B280                      <1>     mov    dl, 80h
3135                                    <1>     ; 15/07/2020
3136 00007766 A2[CD6C0000]              <1>     mov    [hdc], al
3137                                    <1>     ;xor   ecx, ecx ; 0
3138                                    <1> load_next_hd_partition_table:
3139                                    <1>     ; 20/07/2020
3140 0000776B 31C9                      <1>     xor    ecx, ecx ; 0
3141                                    <1>     ;push  ecx
3142 0000776D 57                        <1>     push   edi ; *
3143                                    <1>     ;push  esi ; FDPT (+ DPTE) address
3144                                    <1>     ; 15/07/2020
3145 0000776E AD                        <1>     lodsd
3146 0000776F 56                        <1>     push   esi ; ** ; next FDPT (+ DPTE) address ptr
3147                                    <1>
3148                                    <1>     ;mov   al, [esi+20] ; DPTE offset 4
3149                                    <1>     ;and   al, 40h ;  LBA bit (bit 6)
3150                                    <1>     ;;shr  al, 6
3151                                    <1>     ;mov   [HD_LBA_yes], al
3152                                    <1>
3153                                    <1>     ; 15/07/2020
3154 00007770 8A4814                    <1>     mov    cl, [eax+20]
3155 00007773 80E140                    <1>     and    cl, 40h
3156 00007776 880D[BA8E0100]            <1>     mov    [HD_LBA_yes], cl
3157                                    <1>
3158 0000777C E844040000                <1>     call   load_masterboot
3159                                    <1>     ;jc    short pass_pt_this_hard_disk
3160                                    <1>     ; 13/08/2020
3161 00007781 0F828A000000              <1>     jc     pass_pt_this_hard_disk
3162                                    <1>
3163 00007787 BB[748D0100]              <1>     mov    ebx, PartitionTable
3164 0000778C 89DE                      <1>     mov    esi, ebx
3165                                    <1>     ;mov   ecx, 16
3166 0000778E B110                      <1>     mov    cl, 16
3167 00007790 F3A5                      <1>     rep    movsd
3168 00007792 89DE                      <1>     mov    esi, ebx
3169                                    <1>     ;mov   byte [hdc], 4 ; 4 - partition index
3170                                    <1>     ; 15/07/2020
3171 00007794 C605[BB8E0100]04          <1>     mov    byte [PP_Counter], 4
3172                                    <1> loc_validate_hdp_partition:
3173                                    <1>     ;cmp   byte [esi+ptFileSystemID], 0
3174                                    <1>     ;jna   short loc_validate_next_hdp_partition2
3175                                    <1>     ; 13/08/2020
3176 0000779B 8A4604                    <1>     mov    al, [esi+ptFileSystemID]
3177 0000779E 20C0                      <1>     and    al, al
3178 000077A0 7457                      <1>     jz     short loc_validate_next_hdp_partition2
3179                                    <1>
3180 000077A2 56                        <1>     push   esi ; *** ; Masterboot partition table offset
3181 000077A3 52                        <1>     push   edx ; **** ; dl = Physical drive number
3182                                    <1>
3183                                    <1>     ; 13/08/2020
3184 000077A4 3C05                      <1>     cmp    al, 05h  ; Extended partition CHS
3185 000077A6 7404                      <1>     je     short loc_set_ep_counter
3186 000077A8 3C0F                      <1>     cmp    al, 0Fh  ; Extended partition LBA
3187 000077AA 7511                      <1>     jne    short loc_validate_next_hdp_partition0
3188                                    <1>
3189                                    <1>     ;;inc  byte [PP_Counter]
3190                                    <1>     ; 15/07/2020
3191                                    <1>     ;inc   byte [EP_Counter] ; disk has valid partition(s)
3192                                    <1>
3193                                    <1> loc_set_ep_counter:
3194                                    <1>     ; 13/08/2020
3195 000077AC 803D[BC8E0100]80          <1>     cmp    byte [EP_Counter], 80h
3196 000077B3 7342                      <1>     jnb    short loc_validate_next_hdp_partition1
3197                                    <1>
3198 000077B5 8815[BC8E0100]            <1>     mov    byte [EP_Counter], dl ; disk drv has extd. part.
3199                                    <1>
3200 000077BB EB3A                      <1>     jmp    short loc_validate_next_hdp_partition1
3201                                    <1>
3202                                    <1> loc_validate_next_hdp_partition0:
3203 000077BD 31FF                      <1>     xor    edi, edi ; 0
3204                                    <1>     ; Input -> ESI = PartitionTable offset
3205                                    <1>     ; DL = Hard disk drive number
3206                                    <1>     ; EDI = 0 -> Primary Partition
3207                                    <1>     ; EDI > 0 -> Extended Partition's Start Sector
3208 000077BF E885010000                <1>     call   validate_hd_fat_partition
3209 000077C4 730E                      <1>     jnc    short loc_set_valid_hdp_partition_entry
3210                                    <1>
3211                                    <1>     ;pop   edx
3212                                    <1>     ;push  edx
3213 000077C6 8B1424                    <1>     mov    edx, [esp]  ; ****
3214 000077C9 8B742404                  <1>     mov    esi, [esp+4] ; *** ; 30/01/2018
3215 000077CD E8D1020000                <1>     call   validate_hd_fs_partition
3216 000077D2 7223                      <1>     jc     short loc_validate_next_hdp_partition1
3217                                    <1> loc_set_valid_hdp_partition_entry:
3218 000077D4 8A0D[823F0100]            <1>     mov    cl, [Last_DOS_DiskNo]
3219 000077DA 80C141                    <1>     add    cl, 'A'
3220                                    <1>     ; ESI = Logical dos drive description table address
```

```
3221 000077DD 880E                <1>       mov    [esi+LD_Name], cl
3222                              <1>       ; 15/07/2020
3223 000077DF 8A4602              <1>       mov    al, [esi+LD_PhyDrvNo] ; Physical drive number
3224                              <1>       ;mov   al, [esp] ; ****
3225 000077E2 2C7F                <1>       sub    al, 7Fh
3226                              <1>              ; AL = 1 to 4
3227 000077E4 C0E002              <1>       shl    al, 2 ; AL = 4 to 16
3228                              <1>
3229 000077E7 8A15[BB8E0100]      <1>       mov    dl, [PP_Counter]
3230                              <1>
3231                              <1>       ;sub   al, [PP_Counter]
3232 000077ED 28D0                <1>       sub    al, dl ; [PP_Counter] ; 4 - partition index
3233                              <1>
3234                              <1>       ; AL = Partition entry/index, 0 based
3235                              <1>       ;  0 -> hd 0, Partition Table offset = 0
3236                              <1>       ; 15 -> hd 3, Partition Table offset = 3
3237                              <1>
3238                              <1>       ;mov   [esi+LD_PartitionEntry], al
3239                              <1>
3240                              <1>       ; 15/07/2020
3241 000077EF B404                <1>       mov    ah, 4
3242                              <1>       ;sub   ah, [PP_Counter]
3243 000077F1 28D4                <1>       sub    ah, dl
3244                              <1>
3245                              <1>       ; AH = Primary partition index, 0 to 3 ; pt entry
3246                              <1>       ;            (4 to 7 for logical disk partitions)
3247                              <1>
3248                              <1>       ;mov   [esi+LD_DParamEntry], ah
3249 000077F3 6689467C            <1>       mov    [esi+LD_PartitionEntry], ax
3250                              <1>
3251                              <1> loc_validate_next_hdp_partition1:
3252 000077F7 5A                  <1>       pop    edx ; **** ; dl = Physical drive number
3253 000077F8 5E                  <1>       pop    esi ; *** ; Masterboot partition table offset
3254                              <1>
3255                              <1> loc_validate_next_hdp_partition2:
3256                              <1>       ; ESI = PartitionTable offset
3257                              <1>       ; DL = Hard/Fixed disk drive number
3258                              <1>
3259                              <1>       ;dec   byte [hdc] ; 4 - partition index
3260                              <1>       ;jz    short pass_pt_this_hard_disk
3261                              <1>       ; 15/07/2020
3262 000077F9 FE0D[BB8E0100]      <1>       dec    byte [PP_Counter] ; 4 - partition index
3263 000077FF 7410                <1>       jz     short pass_pt_this_hard_disk
3264                              <1>
3265 00007801 83C610              <1>       add    esi, 16 ; 10h
3266 00007804 EB95                <1>       jmp    short loc_validate_hdp_partition
3267                              <1>
3268                              <1> loc_not_any_extd_partitions:
3269                              <1>       ; 15/07/2020
3270 00007806 C3                  <1>       retn
3271                              <1>
3272                              <1> loc_next_hd_partition_table:
3273 00007807 FEC2                <1>       inc    dl
3274                              <1>       ; 15/07/2020
3275                              <1>       ;add   esi, 32 ; next FDPT address
3276 00007809 83C740              <1>       add    edi, 64 ; next partition table destination
3277 0000780C E95AFFFFFF          <1>         jmp     load_next_hd_partition_table
3278                              <1>
3279                              <1> pass_pt_this_hard_disk:
3280                              <1>       ;pop   esi ; FDPT (+ DPTE) address
3281                              <1>       ; 15/07/2020
3282 00007811 5E                  <1>       pop    esi ; ** ; next FDPT (+ DPTE) address ptr
3283 00007812 5F                  <1>       pop    edi ; * ; Ptable_hd?
3284                              <1>       ;pop   ecx
3285                              <1>       ;loop  loc_next_hd_partition_table
3286 00007813 FE0D[CD6C0000]      <1>       dec    byte [hdc]
3287 00007819 75EC                <1>       jnz    short loc_next_hd_partition_table
3288                              <1>
3289                              <1>       ;cmp   byte [PP_Counter], 1
3290                              <1>       ;jnb   short load_extended_dos_partitions
3291                              <1>       ;; Empty partition table
3292                              <1>       ;retn
3293                              <1>
3294                              <1>       ; 11/08/2020
3295                              <1>       ; 17/07/2020
3296                              <1> check_extended_partitions:
3297                              <1>       ; 15/07/2020
3298                              <1>       ;cmp   byte [EP_Counter], 0
3299                              <1>       ;jna   short loc_not_any_extd_partitions
3300                              <1>       ; 13/08/2020
3301 0000781B A0[BC8E0100]        <1>       mov    al, [EP_Counter] ; 1st disk drv has extd partition
3302 00007820 08C0                <1>       or     al, al ; 0 ?
3303 00007822 74E2                <1>       jz     short loc_not_any_extd_partitions
3304                              <1>
3305                              <1> load_extended_dos_partitions:
3306                              <1>       ;mov   byte [hdc], 80h
3307                              <1>       ; 13/08/2020
3308 00007824 A2[CD6C0000]        <1>       mov    byte [hdc], al ; 1st disk drv has extd partition
3309                              <1>       ; 25/08/2020
3310 00007829 2C80                <1>       sub    al, 80h
3311 0000782B 740E                <1>       jz     short loc_set_ext_ptable_hd0
3312 0000782D C0E006              <1>       shl    al, 6 ; * 64
3313 00007830 0FB6F0              <1>       movzx  esi, al
3314 00007833 81C6[B68D0100]      <1>       add    esi, PTable_hd0
3315 00007839 EB05                <1>       jmp    short next_hd_extd_partition
3316                              <1>
3317                              <1>       ; 25/08/2020
3318                              <1> loc_set_ext_ptable_hd0:
3319 0000783B BE[B68D0100]        <1>       mov    esi, PTable_hd0
3320                              <1>
3321                              <1> next_hd_extd_partition:
3322                              <1>       ; 17/07/2020
3323                              <1>       ;mov   byte [EP_Counter], 0 ; Reset for each physical disk
3324                              <1>       ; 13/08/2020
3325                              <1>       ;mov   byte [LD_Counter], 0 ; Reset logical drive index
```

```
3326 00007840 66C705[BC8E0100]00- <1>       mov    word [EP_Counter], 0 ; Reset EP index and LD index
3326 00007848 00                  <1>
3327                              <1>
3328 00007849 56                  <1>       push   esi ; **** ; PTable_hd? offset
3329                              <1>
3330 0000784A C605[BB8E0100]04    <1>       mov    byte [PP_Counter], 4
3331                              <1>                           ; set for each extd partition table
3332                              <1>       ;;mov  ecx, 4
3333                              <1>       ;mov   cl, 4
3334 00007851 8A15[CD6C0000]      <1>       mov    dl, [hdc]
3335                              <1> hd_check_fs_id_05h:
3336 00007857 8A4604              <1>       mov    al, [esi+ptFileSystemID]
3337 0000785A 3C05                <1>       cmp    al, 05h ; Is it an extended dos partition ?
3338 0000785C 7411                <1>       je     short loc_set_ep_start_sector ; yes
3339                              <1> hd_check_fs_id_0Fh:
3340 0000785E 3C0F                <1>       cmp    al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
3341 00007860 740D                <1>       je     short loc_set_ep_start_sector ; yes
3342                              <1>
3343                              <1> continue_to_check_ep:
3344                              <1>       ;add   esi, 16
3345                              <1>       ;loop  hd_check_fs_id_05h
3346                              <1>       ; 15/07/2020
3347                              <1>       ;dec   cl
3348                              <1>       ;jz    short continue_check_ep_next_disk
3349 00007862 FE0D[BB8E0100]      <1>       dec    byte [PP_Counter] ; 4 --> 0
3350 00007868 742D                <1>       jz     short continue_check_ep_next_disk
3351 0000786A 83C610              <1>       add    esi, 16
3352 0000786D EBE8                <1>       jmp    short hd_check_fs_id_05h
3353                              <1>
3354                              <1> loc_set_ep_start_sector:
3355                              <1>       ; dl = [hdc] ; Drive number
3356                              <1>       ; 15/07/2020
3357 0000786F 8B4E08              <1>       mov    ecx, [esi+ptStartSector]
3358                              <1>       ; 30/08/2020
3359 00007872 890D[BE8E0100]      <1>       mov    [MBR_EP_StartSector], ecx
3360                              <1>       ; 20/07/2020
3361                              <1> loc_validate_hde_partition_next:
3362 00007878 890D[C28E0100]      <1>       mov    [EP_StartSector], ecx ; Extended partition's start sector
3363 0000787E BB[B68B0100]        <1>         mov  ebx, MasterBootBuff
3364 00007883 803D[BA8E0100]01    <1>       cmp    byte [HD_LBA_yes], 1 ; LBA ready = Yes
3365 0000788A 7227                <1>       jb     short loc_hd_load_ep_05h ; cf = 1 ; 20/07/2020
3366                              <1>       ; 11/08/2020
3367                              <1>       ; (BugFix for extended partition type 05h beyond CHS limit)
3368                              <1>       ; (Infact if extended partition starts at the beyond of CHS limit,
3369                              <1>       ;  it's partition ID must be 0Fh but they/somebodies had used 05h.)
3370                              <1>       ;cmp   al, 05h
3371                              <1>       ;je    short loc_hd_load_ep_05h
3372                              <1> loc_hd_load_ep_0Fh:
3373                              <1>       ; 04/01/2016
3374                              <1>       ;push  ecx
3375                              <1>       ; 15/07/2020
3376                              <1>       ;mov   ecx, [esi+ptStartSector] ; sector number
3377                              <1>       ;mov   ebx, MasterBootBuff ; buffer address
3378                              <1>       ; LBA read/write (with private LBA function)
3379                              <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
3380                              <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
3381                              <1>       ;mov   ah, 1Bh ; LBA read
3382                              <1>       ;mov   al, 1 ; sector count
3383 0000788C 66B8011B            <1>       mov    ax, 1B01h
3384 00007890 E8F4D9FFFF          <1>       call   int13h
3385                              <1>       ;pop   ecx
3386                              <1>       ;jnc   short loc_hd_move_ep_table
3387                              <1>       ; 15/07/2020
3388 00007895 732E                <1>       jnc    short loc_validate_hde_partition
3389                              <1>
3390                              <1> continue_check_ep_next_disk:
3391                              <1>       ; 15/07/2020
3392                              <1>       ;pop   edi ; PTable_ep?
3393 00007897 5E                  <1>       pop    esi ; **** ; PTable_hd?
3394 00007898 A0[8C890100]        <1>       mov    al, [HF_NUM] ; number of hard disks
3395 0000789D 047F                <1>       add    al, 7Fh
3396 0000789F 3805[CD6C0000]      <1>       cmp    [hdc], al
3397 000078A5 730B                <1>       jnb    short loc_validating_hd_partitions_ok
3398 000078A7 83C640              <1>       add    esi, 64
3399                              <1>       ; 15/07/2020
3400                              <1>       ;add   edi, 64
3401 000078AA FE05[CD6C0000]      <1>       inc    byte [hdc]
3402 000078B0 EB8E                <1>       jmp    short next_hd_extd_partition
3403                              <1>
3404                              <1> loc_validating_hd_partitions_ok:
3405                              <1>       ; 15/07/2020
3406                              <1>       ;mov   al, [Last_DOS_DiskNo]
3407                              <1> loc_drv_init_retn:
3408 000078B2 C3                  <1>       retn
3409                              <1>
3410                              <1> loc_hd_load_ep_05h:
3411                              <1>       ; 20/07/2020 ('diskio.s', int13h, cf = 1 -> bugfix)
3412                              <1>       ;clc    ; (Bug: int13h would not clear carry flag bit,
3413                              <1>       ;      ; even if there would not be an error)
3414                              <1>       ;      ; ((Fix: now, int13h procedure clears carry flag
3415                              <1>       ;      ;  at the entrance of it.. 20/07/2020))
3416                              <1>       ; 15/07/2020
3417                              <1>       ;push  ecx
3418 000078B3 8A7601              <1>       mov    dh, [esi+ptBeginHead]
3419 000078B6 668B4E02            <1>         mov     cx, [esi+ptBeginSector]
3420 000078BA 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
3421                              <1>       ;mov   ebx, MasterBootBuff
3422 000078BE E8C6D9FFFF          <1>       call   int13h ; 20/07/2020
3423                              <1>                     ; 'diskio.s' modification, 'clc'
3424                              <1>       ;pop   ecx
3425 000078C3 72D2                <1>       jc     short continue_check_ep_next_disk
3426                              <1>       ; 15/07/2020
3427                              <1>       ;jmp   short loc_validate_hde_partition
3428                              <1>
3429                              <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
000078C5 BE[748D0100]       <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
000078CA 807E0400           <1>       cmp    byte [esi+ptFileSystemID], 0
000078CE 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
000078D0 FE05[BC8E0100]     <1>       inc    byte [EP_Counter] ; current (sub partition) index
                            <1>                                ; in current extended partition
                            <1>
000078D6 BF[C28E0100]       <1>       mov    edi, EP_StartSector
                            <1>
                            <1>       ; Input -> ESI = PartitionTable offset
                            <1>       ; DL = Hard disk drive number
                            <1>       ; EDI = Extended partition start sector pointer
000078DB E869000000         <1>       call   validate_hd_fat_partition
                            <1>       ;pop   ecx ; *
000078E0 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
000078E2 8A15[CD6C0000]     <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
000078E8 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
000078EA A0[CD6C0000]       <1>       mov    al, [hdc] ; Hard disk drive number (>=80h)
000078EF 88C2               <1>       mov    dl, al ; mov dl, [hdc]
000078F1 2C7F               <1>       sub    al, 7Fh
                            <1>                  ; 1 to 4
000078F3 C0E002             <1>       shl    al, 2 ; 4 to 16
000078F6 2A05[BB8E0100]     <1>       sub    al, [PP_Counter] ; al - (4 - partition index)
                            <1>                    ; (disk number * 4) + partition index
                            <1>
                            <1>       ; AL = Partition entry/index, 0 based
```

```
3535                                    <1>         ;  0 -> hd 0, Partition Table offset = 0
3536                                    <1>         ; 15 -> hd 3, Partition Table offset = 3
3537                                    <1>
3538                                    <1>       ;mov   ah, 4 ; Logical dos partition (>= 4)
3539                                    <1>       ;add   ah, [EP_Counter]
3540                                    <1>              ; Logical disk partition index = 4 to 7
3541                                    <1>              ; (Primary disk partition index = 0 to 3)
3542                                    <1>
3543                                    <1>       ; 13/08/2020
3544 000078FC 8A25[BD8E0100]           <1>       mov    ah, [LD_Counter] ; Logical drive index number
3545                                    <1>                               ; (in current extended partition)
3546 00007902 80C404                   <1>       add    ah, 4 ; 4 to 7
3547                                    <1>
3548                                    <1>       ; 15/07/2020
3549                                    <1>       ; CX -> AX
3550                                    <1>       ;; 06/01/2016 (TRDOS v2.0)
3551                                    <1>       ;; BUGFIX *
3552                                    <1>       ;;mov  [esi+LD_PartitionEntry], cl
3553                                    <1>       ;;mov  [esi+LD_DParamEntry], ch
3554                                    <1>       ;mov   [esi+LD_PartitionEntry], cx
3555 00007905 6689467C                 <1>       mov    [esi+LD_PartitionEntry], ax
3556                                    <1>
3557 00007909 8A0D[823F0100]           <1>       mov    cl, [Last_DOS_DiskNo]
3558 0000790F 80C141                   <1>       add    cl, 'A'
3559 00007912 880E                     <1>       mov    [esi+LD_Name], cl
3560                                    <1>
3561                                    <1>       ; 17/07/2020
3562                                    <1>       ;cmp   cl, 'Z'
3563                                    <1>       ;jb    short logical_drive_count_ok_for_next
3564                                    <1>       ;pop   esi ; ***
3565                                    <1>       ;pop   esi ; ****
3566                                    <1>       ;retn
3567                                    <1>
3568                                    <1> ;logical_drive_count_ok_for_next:
3569                                    <1>
3570                                    <1>       ;; 15/07/2020
3571                                    <1>       ;inc   byte [EP_Counter]
3572                                    <1>       ; 13/08/2020
3573 00007914 FE05[BD8E0100]           <1>       inc    byte [LD_Counter]
3574                                    <1>
3575                                    <1>       ;mov   dl, [hdc]
3576                                    <1>
3577                                    <1>       ; 17/07/2020
3578                                    <1>       ;; Now,
3579                                    <1>       ;; we are swimming in deep of an extended partition !!!
3580                                    <1>       ; (! sub or chained extended partition tables !)
3581                                    <1>       ; ((Logical dos partitions in extended partition were called
3582                                    <1>       ;  as 'mini disk partition' in msdos 6.0 source code.))
3583                                    <1>
3584                                    <1> validate_next_minidisk_partition:
3585                                    <1>       ; 13/08/2020
3586                                    <1>       ;pop   esi ; *** ; Extended partition table offset
3587                                    <1>
3588                                    <1>       ; 17/07/2020
3589                                    <1>       ;cmp   byte [EP_Counter], 4
3590                                    <1>       ; 13/08/2020
3591 0000791A 803D[BD8E0100]04         <1>       cmp    byte [LD_Counter], 4 ; maximum 4 logical disks
3592                                    <1>                                 ; per extended partition
3593 00007921 0F8370FFFFFF             <1>       jnb    continue_check_ep_next_disk
3594                                    <1>
3595                                    <1> validate_next_minidisk_partition_ok:
3596                                    <1>       ; 13/08/2020
3597                                    <1>       ;dec   byte [PP_Counter] ; 4 --> 0
3598                                    <1>       ;jz    continue_check_ep_next_disk
3599                                    <1>
3600                                    <1>       ;cmp   esi, PartitionTable+64
3601                                    <1>       ;jnb   continue_check_ep_next_disk
3602                                    <1>
3603                                    <1>       ;add   esi, 16
3604                                    <1>       ; 13/08/2020
3605 00007927 BE[848D0100]             <1>       mov    esi, PartitionTable+16
3606                                    <1>
3607                                    <1>       ; 20/07/2020
3608 0000792C 8A4604                   <1>       mov    al, [esi+ptFileSystemID]
3609                                    <1>
3610                                    <1>       ; 20/07/2020
3611 0000792F 3C05                     <1>       cmp    al, 05h ; Is it an extended dos partition ?
3612 00007931 7408                     <1>       je     short loc_minidisk_next_ep_lba_chs ; 17/07/2020
3613 00007933 3C0F                     <1>       cmp    al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
3614 00007935 0F855CFFFFFF             <1>       jne    continue_check_ep_next_disk ; AL must be 0 here
3615                                    <1>                                        ; (when it is not 05h or 0Fh)
3616                                    <1>                                        ; If AL is not ZERO -> EP Bug!
3617                                    <1>                                        ; (!Microsoft DOS convention!)
3618                                    <1> loc_minidisk_next_ep_lba_chs:
3619                                    <1>       ; 17/07/2020
3620 0000793B 8B4E08                   <1>       mov    ecx, [esi+ptStartSector] ; relative start sector number
3621                                    <1>       ;add   ecx, [EP_StartSector]
3622                                    <1>       ; 30/08/2020
3623 0000793E 030D[BE8E0100]           <1>       add    ecx, [MBR_EP_StartSector]
3624                                    <1>       ; 20/07/2020
3625 00007944 E92FFFFFFF               <1>       jmp    loc_validate_hde_partition_next
3626                                    <1>
3627                                    <1> validate_hd_fat_partition:
3628                                    <1>       ; 17/07/2020
3629                                    <1>       ; 15/07/2020
3630                                    <1>       ;      (optimization)
3631                                    <1>       ; 14/07/2020
3632                                    <1>       ;      (fat16 -big- partition search bugfix)
3633                                    <1>       ; 27/12/2017
3634                                    <1>       ; 12/02/2016
3635                                    <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
3636                                    <1>       ; 07/08/2011
3637                                    <1>       ; 23/07/2011
3638                                    <1>       ; Input
3639                                    <1>       ;   DL = Hard/Fixed Disk Drive Number
```

```
3640                                    <1>       ;   ESI = PartitionTable offset
3641                                    <1>       ;   EDI = Extend. Part. Start Sector Pointer
3642                                    <1>       ;   EDI = 0 -> Primary Partition
3643                                    <1>       ;   byte [Last_DOS_DiskNo]
3644                                    <1>       ; Output
3645                                    <1>       ;  cf=0 -> Validated
3646                                    <1>       ;   ESI = Logical dos drv desc. table
3647                                    <1>       ;   EBX = FAT boot sector buffer
3648                                    <1>       ;   byte [Last_DOS_DiskNo]
3649                                    <1>       ;  cf=1 -> Not a valid FAT partition
3650                                    <1>       ; EAX, EDX, ECX, EDI -> changed
3651                                    <1>
3652                                    <1>       ;mov   esi, PartitionTable
3653 00007949 8A6604                   <1>       mov    ah, [esi+ptFileSystemID]
3654 0000794C B002                     <1>       mov    al, 2 ; 27/12/2017
3655 0000794E 80FC06                   <1>       cmp    ah, 06h ; FAT16 CHS partition (>=32MB)
3656                                    <1>       ; 12/02/2016
3657                                    <1>       ;;jb   short loc_not_a_valid_fat_partition2
3658                                    <1>       ;jnb   short vhdp_FAT16_32
3659                                    <1>       ; 14/07/2020 (BugFix)
3660 00007951 7711                     <1>       ja     short vhdp_FAT16_32
3661 00007953 7425                     <1>       je     short loc_set_valid_hd_partition_params
3662                                    <1>
3663                                    <1> vhdp_FAT12_16:
3664                                    <1>       ; 27/12/2017
3665 00007955 FEC8                     <1>       dec    al ; mov al, 1
3666 00007957 38C4                     <1>       cmp    ah, al ; 1 ; FAT12 partition
3667 00007959 741F                     <1>       je     short loc_set_valid_hd_partition_params
3668                                    <1>       ;
3669 0000795B FEC0                     <1>       inc    al ; mov al, 2
3670 0000795D 80FC04                   <1>       cmp    ah, 04h ; FAT16 CHS partition (< 32MB)
3671 00007960 7418                     <1>       je     short loc_set_valid_hd_partition_params
3672                                    <1>
3673                                    <1>       ; 15/07/2020
3674                                    <1>       ; (ah = 05h, 02h or 03h)
3675                                    <1> loc_not_a_valid_fat_partition1:
3676 00007962 F9                       <1>       stc
3677                                    <1>       ; cf=1
3678 00007963 C3                       <1>       retn
3679                                    <1>
3680                                    <1> vhdp_FAT16_32:
3681                                    <1>       ; 15/07/2020
3682                                    <1>       ;mov   al, 3
3683 00007964 FEC0                     <1>       inc    al
3684 00007966 80FC0C                   <1>       cmp    ah, 0Ch ; FAT32 LBA partition
3685 00007969 740F                     <1>       je     short loc_set_valid_hd_partition_params
3686 0000796B 7706                     <1>       ja     short vhdp_check_FAT16_lba
3687                                    <1>
3688                                    <1> vhdp_check_FAT32_chs:
3689 0000796D 80FC0B                   <1>       cmp    ah, 0Bh ; FAT32 CHS partition
3690 00007970 7408                     <1>       je     short loc_set_valid_hd_partition_params
3691                                    <1>       ;jne   short loc_not_a_valid_fat_partition1
3692                                    <1>
3693                                    <1>       ;stc
3694                                    <1> loc_not_a_valid_fat_partition2:
3695 00007972 C3                       <1>       retn
3696                                    <1>
3697                                    <1> vhdp_check_FAT16_lba:
3698 00007973 80FC0E                   <1>       cmp    ah, 0Eh ; FAT16 LBA partition
3699 00007976 75EA                     <1>       jne    short loc_not_a_valid_fat_partition1
3700                                    <1>
3701                                    <1>       ;mov   al, 2
3702 00007978 FEC8                     <1>       dec    al
3703                                    <1>
3704                                    <1> loc_set_valid_hd_partition_params:
3705                                    <1>       ; 15/07/2020
3706                                    <1>       ;inc   byte [Last_DOS_DiskNo] ; > 1
3707                                    <1>       ;
3708 0000797A 31DB                     <1>       xor    ebx, ebx
3709 0000797C 8A3D[823F0100]           <1>       mov    bh, [Last_DOS_DiskNo] ; * 256
3710 00007982 FEC7                     <1>       inc    bh ; 15/07/2020
3711 00007984 81C300010900             <1>       add    ebx, Logical_DOSDisks
3712                                    <1>       ;
3713 0000798A C6430102                 <1>       mov    byte [ebx+LD_DiskType], 2
3714 0000798E 885302                   <1>       mov    byte [ebx+LD_PhyDrvNo], dl
3715                                    <1>       ;mov   byte [ebx+LD_FATType], al ; 2 or 3
3716                                    <1>       ;mov   byte [ebx+LD_FSType], ah ; 06h, 0Eh, 0Bh, 0Ch
3717 00007991 66894303                 <1>       mov    word [ebx+LD_FATType], ax
3718                                    <1>       ;
3719 00007995 8B4E08                   <1>       mov    ecx, [esi+ptStartSector]
3720 00007998 09FF                     <1>       or     edi, edi
3721 0000799A 7402                     <1>       jz     short pass_hd_FAT_ep_start_sector_adding
3722                                    <1> loc_add_hd_FAT_ep_start_sector:
3723                                    <1>       ; 17/07/2020
3724 0000799C 030F                     <1>       add    ecx, [edi]
3725                                    <1> pass_hd_FAT_ep_start_sector_adding:
3726 0000799E 894B6C                   <1>       mov    [ebx+LD_StartSector], ecx
3727                                    <1> loc_hd_FAT_logical_drv_init:
3728 000079A1 89DD                     <1>       mov    ebp, ebx
3729                                    <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
3730 000079A3 A0[BA8E0100]             <1>       mov    al, [HD_LBA_yes] ; 07/01/2016
3731 000079A8 884305                   <1>       mov    [ebx+LD_LBAYes], al
3732 000079AB BB[C68E0100]             <1>       mov    ebx, DOSBootSectorBuff ; buffer address
3733 000079B0 08C0                     <1>       or     al, al
3734 000079B2 740C                     <1>       jz     short loc_hd_FAT_drv_init_load_bs_chs
3735                                    <1> loc_hd_FAT_drv_init_load_bs_lba:
3736                                    <1>       ; DL = Physical drive number
3737                                    <1>       ;mov   ecx, [esi+ptStartSector] ; sector number
3738                                    <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
3739                                    <1>       ; LBA read/write (with private LBA function)
3740                                    <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
3741                                    <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
3742 000079B4 B41B                     <1>       mov    ah, 1Bh ; LBA read
3743 000079B6 B001                     <1>       mov    al, 1 ; sector count
3744 000079B8 E8CCD8FFFF               <1>       call   int13h
```

```
3745 000079BD 7313                <1>       jnc    short loc_hd_drv_FAT_boot_validation
3746                              <1> loc_not_a_valid_fat_partition3:
3747 000079BF C3                  <1>       retn
3748                              <1> loc_hd_FAT_drv_init_load_bs_chs:
3749 000079C0 8A7601              <1>       mov    dh, [esi+ptBeginHead]
3750 000079C3 668B4E02            <1>       mov    cx, [esi+ptBeginSector]
3751 000079C7 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
3752                              <1>       ;mov   ebx, DOSBootSectorBuff
3753 000079CB E8B9D8FFFF          <1>       call   int13h
3754 000079D0 72ED                <1>       jc     short loc_not_a_valid_fat_partition3
3755                              <1> loc_hd_drv_FAT_boot_validation:
3756                              <1>       ;mov   esi, DOSBootSectorBuff
3757 000079D2 89DE                <1>       mov    esi, ebx
3758 000079D4 6681BEFE01000055AA  <1>       cmp    word [esi+BS_Validation], 0AA55h
3759 000079DD 7512                <1>       jne    short loc_not_a_valid_fat_partition4
3760 000079DF 807E15F8            <1>       cmp    byte [esi+BPB_Media], 0F8h
3761 000079E3 750C                <1>       jne    short loc_not_a_valid_fat_partition4
3762                              <1>
3763                              <1>       ; 27/12/2017
3764 000079E5 807D0303            <1>       cmp    byte [ebp+LD_FATType], 3
3765 000079E9 7508                <1>       jne    short loc_hd_FAT16_BPB
3766                              <1>
3767                              <1> loc_hd_drv_FAT32_boot_validation:
3768 000079EB 807E4229            <1>       cmp    byte [esi+BS_FAT32_BootSig], 29h
3769 000079EF 7416                <1>       je     short loc_hd_FAT32_BPB
3770                              <1>
3771                              <1> loc_not_a_valid_fat_partition4:
3772 000079F1 F9                  <1>       stc
3773 000079F2 C3                  <1>       retn
3774                              <1>
3775                              <1> loc_hd_FAT16_BPB:
3776 000079F3 807E2629            <1>       cmp    byte [esi+BS_BootSig], 29h
3777 000079F7 75F8                <1>       jne    short loc_not_a_valid_fat_partition4
3778                              <1>
3779 000079F9 66837E1600          <1>       cmp    word [esi+BPB_FATSz16], 0
3780 000079FE 7607                <1>       jna    short loc_hd_big_FAT16_BPB
3781 00007A00 B920000000          <1>       mov    ecx, 32
3782 00007A05 EB05                <1>       jmp    short loc_hd_move_FAT_BPB
3783                              <1>
3784                              <1> loc_hd_FAT32_BPB:
3785                              <1>       ;cmp   word [esi+BPB_FATSz16], 0
3786                              <1>       ;ja    short loc_not_a_valid_fat_partition4
3787                              <1> loc_hd_big_FAT16_BPB:
3788 00007A07 B92D000000          <1>       mov    ecx, 45
3789                              <1>
3790                              <1> loc_hd_move_FAT_BPB:
3791 00007A0C 89EF                <1>       mov    edi, ebp
3792                              <1>       ;mov   esi, ebx ; Boot sector
3793 00007A0E 57                  <1>       push   edi
3794 00007A0F 83C706              <1>       add    edi, LD_BPB
3795 00007A12 F366A5              <1>       rep    movsw
3796 00007A15 5E                  <1>       pop    esi
3797 00007A16 0FB74614            <1>       movzx  eax, word [esi+LD_BPB+BPB_RsvdSecCnt]
3798 00007A1A 03466C              <1>       add    eax, [esi+LD_StartSector]
3799 00007A1D 894660              <1>       mov    [esi+LD_FATBegin], eax
3800 00007A20 807E0303            <1>       cmp    byte [esi+LD_FATType], 3
3801 00007A24 7224                <1>       jb     short loc_set_FAT16_RootDirLoc
3802                              <1> loc_set_FAT32_RootDirLoc:
3803 00007A26 8B462A              <1>       mov    eax, [esi+LD_BPB+BPB_FATSz32]
3804 00007A29 0FB65E16            <1>         movzx      ebx, byte [esi+LD_BPB+BPB_NumFATs]
3805 00007A2D F7E3                <1>       mul    ebx
3806 00007A2F 034660              <1>       add    eax, [esi+LD_FATBegin]
3807                              <1> loc_set_FAT32_data_begin:
3808 00007A32 894668              <1>       mov    [esi+LD_DATABegin], eax
3809 00007A35 894664              <1>       mov    [esi+LD_ROOTBegin], eax
3810                              <1>       ; If Root Directory Cluster <> 2 then
3811                              <1>       ; change the beginning sector value
3812                              <1>       ; of the root dir by adding sector offset.
3813 00007A38 8B4632              <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
3814 00007A3B 83E802              <1>       sub    eax, 2
3815 00007A3E 7435                <1>       jz     short short loc_set_32bit_FAT_total_sectors
3816                              <1>       ;movzx ebx, byte [esi+LD_BPB+BPB_SecPerClust]
3817 00007A40 8A5E13              <1>       mov    bl, [esi+LD_BPB+BPB_SecPerClust]
3818 00007A43 F7E3                <1>       mul    ebx
3819 00007A45 014664              <1>       add    [esi+LD_ROOTBegin], eax
3820 00007A48 EB2B                <1>       jmp    short loc_set_32bit_FAT_total_sectors
3821                              <1>       ;
3822                              <1> loc_set_FAT16_RootDirLoc:
3823 00007A4A 0FB64616            <1>       movzx  eax, byte [esi+LD_BPB+BPB_NumFATs]
3824 00007A4E 0FB7561C            <1>       movzx  edx, word [esi+LD_BPB+BPB_FATSz16]
3825 00007A52 F7E2                <1>       mul    edx
3826 00007A54 034660              <1>       add    eax, [esi+LD_FATBegin]
3827 00007A57 894664              <1>       mov    [esi+LD_ROOTBegin], eax
3828                              <1> loc_set_FAT16_data_begin:
3829 00007A5A 894668              <1>       mov    [esi+LD_DATABegin], eax
3830                              <1>       ;mov   eax, 20h  ; Size of a directory entry
3831                              <1>       ;;movzx      edx, word [esi+LD_BPB+BPB_RootEntCnt]
3832                              <1>         ;mov     dx, [esi+LD_BPB+BPB_RootEntCnt]
3833                              <1>         ;mul edx
3834                              <1>       ;;mov  ecx, 511
3835                              <1>       ;mov   cx, 511
3836                              <1>       ;add   eax, ecx
3837                              <1>       ;inc   ecx ; 512
3838                              <1>       ;div   ecx
3839                              <1>       ; 14/07/2020
3840 00007A5D 0FB74617            <1>       movzx  eax, word [esi+LD_BPB+BPB_RootEntCnt]
3841 00007A61 6683C00F            <1>       add    ax, 15
3842 00007A65 66C1E804            <1>       shr    ax, 4 ; / 16 ; (16 entries per sector)
3843 00007A69 014668              <1>       add    [esi+LD_DATABegin], eax
3844                              <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
3845 00007A6C 668B4619            <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
3846 00007A70 6685C0              <1>       test   ax, ax
3847                              <1>       ;jz    short loc_set_32bit_FAT_total_sectors
3848                              <1> ;loc_set_16bit_FAT_total_sectors:
3849                              <1>       ;mov   [esi+LD_TotalSectors], eax
```

```
                        <1>       ;jmp   short loc_set_hd_FAT_cluster_count
                        <1>       ; 14/07/2020
00007A73 7503           <1>       jnz    short loc_set_hd_FAT_cluster_count
                        <1> loc_set_32bit_FAT_total_sectors:
00007A75 8B4626         <1>       mov    eax, [esi+LD_BPB+BPB_TotalSec32]
                        <1>       ;mov   [esi+LD_TotalSectors], eax
                        <1> loc_set_hd_FAT_cluster_count:
00007A78 894670         <1>       mov    [esi+LD_TotalSectors], eax ; 14/07/2020
00007A7B 8B5668         <1>       mov    edx, [esi+LD_DATABegin]
00007A7E 2B566C         <1>       sub    edx, [esi+LD_StartSector]
00007A81 29D0           <1>       sub    eax, edx
00007A83 31D2           <1>       xor    edx, edx ; 0
00007A85 0FB64E13       <1>         movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
00007A89 F7F1           <1>         div ecx
00007A8B 894678         <1>       mov    [esi+LD_Clusters], eax
                        <1>       ; Maximum Valid Cluster Number= EAX +1
                        <1>       ; with 2 reserved clusters= EAX +2
                        <1> loc_set_hd_FAT_fs_free_sectors:
                        <1>       ;mov   dword [esi+LD_FreeSectors], 0
00007A8E E855010000     <1>       call   get_free_FAT_sectors
00007A93 720D           <1>       jc     short loc_validate_hd_FAT_partition_retn
00007A95 894674         <1>       mov    [esi+LD_FreeSectors], eax
00007A98 C6467E06       <1>       mov    byte [esi+LD_MediaChanged], 6  ; Volume Name Reset
                        <1>
                        <1>       ; 15/07/2020
00007A9C FE05[823F0100] <1>       inc    byte [Last_DOS_DiskNo] ; > 1
                        <1>
                        <1>       ;mov   cl, [Last_DOS_DiskNo]
                        <1>       ;add   cl, 'A'
                        <1>       ;mov   [esi+LD_FS_Name], cl
                        <1>
                        <1> loc_validate_hd_FAT_partition_retn:
00007AA2 C3             <1>       retn
                        <1>
                        <1> validate_hd_fs_partition:
                        <1>       ; 03/02/2018
                        <1>       ; 09/12/2017
                        <1>       ; 13/02/2016
                        <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 29/01/2011
                        <1>       ; 23/07/2011
                        <1>       ; Input
                        <1>       ;   DL = Hard/Fixed Disk Drive Number
                        <1>       ;   ESI = PartitionTable offset
                        <1>       ;   byte [Last_DOS_DiskNo]
                        <1>       ; Output
                        <1>       ;  cf=0 -> Validated
                        <1>       ;   ESI = Logical dos drv desc. table
                        <1>       ;   EBX = Singlix FS boot sector buffer
                        <1>       ;   byte [Last_DOS_DiskNo]
                        <1>       ;  cf=1 -> Not a valid 'Singlix FS' partition
                        <1>       ; EAX, EDX, ECX, EDI -> changed
                        <1>
                        <1>       ;mov   esi, PartitionTable
00007AA3 8A6604         <1>       mov    ah, [esi+ptFileSystemID]
00007AA6 80FCA1         <1>       cmp    ah, 0A1h ; SINGLIX FS1 (trfs1) partition
00007AA9 7549           <1>       jne    short loc_validate_hd_fs_partition_stc_retn
                        <1> loc_set_valid_hd_fs_partition_params:
00007AAB FE05[823F0100] <1>       inc    byte [Last_DOS_DiskNo] ; > 1
00007AB1 30C0           <1>       xor    al, al ; mov al, 0
                        <1>       ;mov   [drv], dl
00007AB3 29DB           <1>       sub    ebx, ebx ; 0
00007AB5 8A3D[823F0100] <1>       mov    bh, [Last_DOS_DiskNo]
00007ABB 81C300010900   <1>       add    ebx, Logical_DOSDisks
00007AC1 C6430102       <1>       mov    byte [ebx+LD_DiskType], 2
00007AC5 885302         <1>       mov    [ebx+LD_PhyDrvNo], dl
                        <1>       ;mov   [ebx+LD_FATType], al ; 0
                        <1>       ;mov   [ebx+LD_FSType], ah
00007AC8 66894303       <1>       mov    [ebx+LD_FATType], ax
                        <1>       ;mov   eax, [esi+ptStartSector]
                        <1>       ;mov   [ebx+LD_StartSector], eax
                        <1> loc_hd_fs_logical_drv_init:
00007ACC 89DD           <1>       mov    ebp, ebx ; 10/01/2016
                        <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
00007ACE A0[BA8E0100]   <1>       mov    al, [HD_LBA_yes] ; 10/01/2016
00007AD3 884305         <1>       mov    [ebx+LD_LBAYes], al
00007AD6 89DE           <1>       mov    esi, ebx
00007AD8 BB[C68E0100]   <1>       mov    ebx, DOSBootSectorBuff ; buffer address
00007ADD 08C0           <1>       or     al, al
00007ADF 7515           <1>       jnz    short loc_hd_fs_drv_init_load_bs_lba
                        <1> loc_hd_fs_drv_init_load_bs_chs:
00007AE1 8A7601         <1>       mov    dh, [esi+ptBeginHead]
00007AE4 668B4E02       <1>       mov    cx, [esi+ptBeginSector]
00007AE8 66B80102       <1>       mov    ax, 0201h ; Read 1 sector
                        <1>       ;mov   ebx, DOSBootSectorBuff
00007AEC E898D7FFFF     <1>       call   int13h
00007AF1 7311           <1>       jnc    short loc_hd_drv_fs_boot_validation
                        <1> loc_validate_hd_fs_partition_err_retn:
00007AF3 C3             <1>       retn
                        <1> loc_validate_hd_fs_partition_stc_retn:
00007AF4 F9             <1>       stc
00007AF5 C3             <1>       retn
                        <1> loc_hd_fs_drv_init_load_bs_lba:
                        <1>       ; DL = Physical drive number
                        <1>       ;mov   esi, ebx
00007AF6 8B4E08         <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)
00007AF9 B41B           <1>       mov    ah, 1Bh ; LBA read
00007AFB B001           <1>       mov    al, 1 ; sector count
00007AFD E887D7FFFF     <1>       call   int13h
00007B02 72EF           <1>       jc     short loc_validate_hd_fs_partition_err_retn
                        <1> loc_hd_drv_fs_boot_validation:
```

```
3955                                    <1>         ;mov   esi, DOSBootSectorBuff
3956 00007B04 89DE                     <1>         mov    esi, ebx ; Boot sector buffer
3957 00007B06 6681BEFE01000055AA       <1>         cmp    word [esi+BS_Validation], 0AA55h
3958 00007B0F 75E3                     <1>         jne    short loc_validate_hd_fs_partition_stc_retn
3959                                    <1>           ;
3960                                    <1>         ;Singlix FS Extensions to TR-DOS (7/6/2009)
3961 00007B11 66817E034653             <1>         cmp    word [esi+bs_FS_Identifier], 'FS' ; 03/02/2018
3962 00007B17 75DB                     <1>         jne    short loc_validate_hd_fs_partition_stc_retn
3963                                    <1>           ;'A1h' check is not necessary
3964                                    <1>         ;  if 'FS' check is passed as OK/Yes.
3965 00007B19 807E09A1                 <1>         cmp    byte [esi+bs_FS_PartitionID], 0A1h
3966 00007B1D 75D5                     <1>         jne    short loc_validate_hd_fs_partition_stc_retn
3967                                    <1>         ;
3968 00007B1F 89EF                     <1>         mov    edi, ebp ; 10/01/2016
3969                                    <1>         ;
3970 00007B21 8A462D                   <1>         mov    al, byte [esi+bs_FS_LBA_Ready]
3971 00007B24 884705                   <1>         mov    [edi+LD_FS_LBAYes], al
3972                                    <1>         ;
3973                                    <1>         ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
3974 00007B27 8A4608                   <1>         mov    al, [esi+bs_FS_MediaAttrib]
3975 00007B2A 884706                   <1>         mov    byte [edi+LD_FS_MediaAttrib], al
3976                                    <1>         ;
3977 00007B2D 8A460A                   <1>         mov    al, [esi+bs_FS_VersionMaj]
3978 00007B30 884707                   <1>         mov    [edi+LD_FS_VersionMajor], al
3979                                    <1>         ;
3980 00007B33 668B4606                 <1>         mov    ax, [esi+bs_FS_BytesPerSec]
3981 00007B37 66894711                 <1>         mov    [edi+LD_FS_BytesPerSec], ax
3982 00007B3B 8A462E                   <1>         mov    al, [esi+bs_FS_SecPerTrack]
3983 00007B3E 30E4                     <1>         xor    ah, ah ; 09/12/2017
3984 00007B40 6689471E                 <1>         mov    [edi+LD_FS_SecPerTrack], ax
3985 00007B44 8A462F                   <1>         mov    al, [esi+bs_FS_Heads]
3986 00007B47 66894720                 <1>         mov    [edi+LD_FS_NumHeads], ax
3987                                    <1>         ;
3988 00007B4B 8B4628                   <1>         mov    eax, [esi+bs_FS_UnDelDirD]
3989 00007B4E 894722                   <1>         mov    [edi+LD_FS_UnDelDirD], eax
3990 00007B51 8B5618                   <1>         mov    edx, [esi+bs_FS_MATLocation]
3991 00007B54 89570C                   <1>         mov    [edi+LD_FS_MATLocation], edx
3992 00007B57 8B461C                   <1>         mov    eax, [esi+bs_FS_RootDirD]
3993 00007B5A 894708                   <1>         mov    [edi+LD_FS_RootDirD], eax
3994 00007B5D 8B460C                   <1>         mov    eax, [esi+bs_FS_BeginSector]
3995 00007B60 89476C                   <1>         mov    [edi+LD_FS_BeginSector], eax
3996 00007B63 8B4710                   <1>         mov    eax, [edi+bs_FS_VolumeSize]
3997 00007B66 894770                   <1>         mov    [edi+LD_FS_VolumeSize], eax
3998                                    <1>         ;
3999 00007B69 89D0                     <1>         mov    eax, edx ; [edi+LD_FS_MATLocation]
4000 00007B6B 03476C                   <1>         add    eax, [edi+LD_FS_BeginSector]
4001 00007B6E 89FE                     <1>         mov    esi, edi
4002                                    <1> mread_hd_fs_MAT_sector:
4003                                    <1>          ;mov  ebx, DOSBootSectorBuff
4004 00007B70 B901000000               <1>         mov    ecx, 1
4005 00007B75 E803AF0000               <1>         call   disk_read
4006 00007B7A 7248                     <1>         jc     short loc_validate_hd_fs_partition_retn
4007                                    <1>         ; EDI will not be changed
4008 00007B7C 89DE                     <1>         mov    esi, ebx
4009                                    <1> use_hdfs_mat_sector_params:
4010 00007B7E 8B460C                   <1>         mov    eax, [esi+FS_MAT_DATLocation]
4011 00007B81 894714                   <1>         mov    [edi+LD_FS_DATLocation], eax
4012 00007B84 8B4610                   <1>         mov    eax, [esi+FS_MAT_DATScount]
4013 00007B87 894718                   <1>         mov    [edi+LD_FS_DATSectors], eax
4014 00007B8A 8B4614                   <1>         mov    eax, [esi+FS_MAT_FreeSectors]
4015 00007B8D 894774                   <1>           mov     [edi+LD_FS_FreeSectors], eax
4016 00007B90 8B4618                   <1>         mov    eax, [esi+FS_MAT_FirstFreeSector]
4017 00007B93 894778                   <1>         mov    [edi+LD_FS_FirstFreeSector], eax
4018 00007B96 8B4708                   <1>         mov    eax, [edi+LD_FS_RootDirD]
4019 00007B99 03476C                   <1>         add    eax, [edi+LD_FS_BeginSector]
4020 00007B9C 89FE                     <1>         mov    esi, edi
4021                                    <1> read_hd_fs_RDT_sector:
4022 00007B9E BB[C68E0100]             <1>         mov    ebx, DOSBootSectorBuff
4023                                    <1>         ;mov   ecx, 1
4024 00007BA3 B101                     <1>         mov    cl, 1
4025 00007BA5 E8D3AE0000               <1>         call   disk_read
4026 00007BAA 7218                     <1>         jc     short loc_validate_hd_fs_partition_retn
4027                                    <1>         ; EDI will not be changed
4028 00007BAC 89DE                     <1>         mov    esi, ebx
4029                                    <1> use_hdfs_RDT_sector_params:
4030 00007BAE 8B461C                   <1>         mov    eax, [esi+FS_RDT_VolumeSerialNo]
4031 00007BB1 894728                   <1>         mov    [edi+LD_FS_VolumeSerial], eax
4032 00007BB4 57                       <1>         push   edi
4033                                    <1>         ;mov   ecx, 16
4034 00007BB5 B110                     <1>         mov    cl, 16
4035 00007BB7 83C640                   <1>         add    esi, FS_RDT_VolumeName
4036 00007BBA 83C72C                   <1>         add    edi, LD_FS_VolumeName
4037 00007BBD F3A5                     <1>         rep    movsd ; 64 bytes
4038 00007BBF 5E                       <1>         pop    esi
4039                                    <1>                ; Volume Name Reset
4040 00007BC0 C6467E06                 <1>           mov     byte [esi+LD_FS_MediaChanged], 6
4041                                    <1>         ;
4042                                    <1>           ;mov cl, [Last_DOS_DiskNo]
4043                                    <1>         ;add   cl, 'A'
4044                                    <1>         ;mov   [esi+LD_FS_Name], cl
4045                                    <1>
4046                                    <1> loc_validate_hd_fs_partition_retn:
4047 00007BC4 C3                       <1>         retn
4048                                    <1>
4049                                    <1> load_masterboot:
4050                                    <1>         ; 14/07/2020 (Reset function has been removed)
4051                                    <1>         ;
4052                                    <1>         ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4053                                    <1>         ; 2005 - 2011
4054                                    <1>         ; input -> DL = drive number
4055                                    <1> ;       mov    ah, 0Dh ; Alternate disk reset
4056                                    <1> ;       call   int13h
4057                                    <1> ;       jnc    short pass_reset_error
4058                                    <1> ;harddisk_error:
4059                                    <1> ;       retn
```

```
4060                                    <1> ;pass_reset_error:
4061 00007BC5 BB[B68B0100]              <1>         mov     ebx, MasterBootBuff
4062 00007BCA 66B80102                  <1>         mov     ax, 0201h
4063 00007BCE 66B90100                  <1>         mov     cx, 1
4064 00007BD2 30F6                      <1>         xor     dh, dh
4065 00007BD4 E8B0D6FFFF                <1>         call    int13h
4066 00007BD9 720C                      <1>         jc      short harddisk_error
4067                                    <1>         ;
4068 00007BDB 66813D[B48D0100]55-       <1>         cmp     word [MBIDCode], 0AA55h
4068 00007BE3 AA                        <1>
4069 00007BE4 7401                      <1>         je      short load_masterboot_ok
4070 00007BE6 F9                        <1>         stc
4071                                    <1> harddisk_error:
4072                                    <1> load_masterboot_ok:
4073 00007BE7 C3                        <1>         retn
4074                                    <1>
4075                                    <1> get_free_FAT_sectors:
4076                                    <1>         ; 21/12/2017
4077                                    <1>         ; 29/02/2016
4078                                    <1>         ; 13/02/2016
4079                                    <1>         ; 04/02/2016
4080                                    <1>         ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
4081                                    <1>         ; 11/07/2010
4082                                    <1>         ; 21/06/2009
4083                                    <1>         ; INPUT: ESI = Logical DOS Drive Description Table address
4084                                    <1>         ; OUTPUT: STC => Error
4085                                    <1>           ;    cf = 0 and EAX = Free FAT sectors
4086                                    <1>         ; Also, related parameters and FAT buffer will be reset and updated
4087                                    <1>
4088 00007BE8 31C0                      <1>         xor     eax, eax
4089                                    <1>         ;mov   [esi+LD_FreeSectors], eax ; Reset
4090                                    <1>
4091 00007BEA 807E0302                  <1>           cmp     byte [esi+LD_FATType], 2
4092 00007BEE 7654                      <1>         jna     short loc_gfc_get_fat_free_clusters
4093                                    <1>
4094                                    <1>         ; 29/02/2016
4095 00007BF0 48                        <1>         dec     eax ; 0FFFFFFFFh
4096 00007BF1 89463A                    <1>         mov     [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count (reset)
4097 00007BF4 89463E                    <1>         mov     [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster (reset)
4098 00007BF7 40                        <1>         inc     eax ; 0
4099                                    <1>         ;
4100 00007BF8 668B4636                  <1>         mov     ax, [esi+LD_BPB+BPB_FSInfo]
4101 00007BFC 03466C                    <1>         add     eax, [esi+LD_StartSector]
4102                                    <1>
4103 00007BFF BB[C68E0100]              <1>         mov     ebx, DOSBootSectorBuff
4104 00007C04 B901000000                <1>         mov     ecx, 1
4105 00007C09 E86FAE0000                <1>         call    disk_read
4106 00007C0E 7301                      <1>         jnc     short loc_gfc_check_fsinfo_signs
4107                                    <1> retn_gfc_get_fsinfo_sec:
4108 00007C10 C3                        <1>         retn
4109                                    <1>
4110                                    <1> loc_gfc_check_fsinfo_signs:
4111 00007C11 BB[C68E0100]              <1>         mov     ebx, DOSBootSectorBuff ; 13/02/2016
4112 00007C16 813B52526141              <1>           cmp     dword [ebx], 41615252h
4113 00007C1C 7524                      <1>         jne     short retn_gfc_get_fsinfo_stc
4114                                    <1>         ;add   ebx, 484
4115                                    <1>         ;cmp   dword [ebx], 61417272h
4116 00007C1E 81BBE4010000727241-       <1>         cmp     dword [ebx+484], 61417272h
4116 00007C27 61                        <1>
4117 00007C28 7518                      <1>         jne     short retn_gfc_get_fsinfo_stc
4118                                    <1>         ;add   ebx, 4
4119                                    <1>         ;mov   eax, [ebx]
4120 00007C2A 8B83E8010000              <1>         mov     eax, [ebx+488]
4121                                    <1>         ; 29/02/2016
4122 00007C30 89463A                    <1>         mov     [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
4123 00007C33 8B93EC010000              <1>         mov     edx,  [ebx+492]
4124 00007C39 89463E                    <1>         mov     [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster
4125                                    <1>         ; 21/12/2017
4126 00007C3C 89C3                      <1>         mov     ebx, eax ; (initial value = 0FFFFFFFFh)
4127 00007C3E 43                        <1>         inc     ebx ; 0FFFFFFFFh -> 0
4128 00007C3F 7513                      <1>         jnz     short short retn_from_get_free_fat32_clusters
4129 00007C41 C3                        <1>         retn
4130                                    <1>
4131                                    <1> retn_gfc_get_fsinfo_stc:
4132 00007C42 F9                        <1>         stc
4133 00007C43 C3                        <1>         retn
4134                                    <1>
4135                                    <1> loc_gfc_get_fat_free_clusters:
4136                                    <1>         ;mov   eax, 2
4137 00007C44 B002                      <1>         mov     al, 2
4138                                    <1>         ;mov   [FAT_CurrentCluster], eax
4139                                    <1> loc_gfc_loop_get_next_cluster:
4140 00007C46 E8EB4F0000                <1>         call    get_next_cluster
4141 00007C4B 730E                      <1>         jnc     short loc_gfc_free_fat_clusters_cont
4142 00007C4D 21C0                      <1>         and     eax, eax
4143 00007C4F 7411                      <1>         jz      short loc_gfc_pass_inc_free_cluster_count
4144                                    <1>
4145                                    <1> retn_from_get_free_fat_clusters:
4146 00007C51 8B4674                    <1>         mov     eax, [esi+LD_FreeSectors] ; Free clusters !
4147                                    <1> retn_from_get_free_fat32_clusters:
4148 00007C54 0FB65E13                  <1>           movzx       ebx, byte [esi+LD_BPB+BPB_SecPerClust]
4149 00007C58 F7E3                      <1>                mul     ebx
4150                                    <1>         ;mov   [esi+LD_FreeSectors], eax ; Free sectors
4151                                    <1> retn_get_free_sectors_calc:
4152 00007C5A C3                        <1>         retn
4153                                    <1>
4154                                    <1> loc_gfc_free_fat_clusters_cont:
4155 00007C5B 09C0                      <1>         or      eax, eax
4156 00007C5D 7503                      <1>         jnz     short loc_gfc_pass_inc_free_cluster_count
4157 00007C5F FF4674                    <1>         inc     dword [esi+LD_FreeSectors] ; Free clusters !
4158                                    <1>
4159                                    <1> loc_gfc_pass_inc_free_cluster_count:
4160                                    <1>         ;mov   eax, [FAT_CurrentCluster]
4161 00007C62 89C8                      <1>         mov     eax, ecx ; [FAT_CurrentCluster]
4162 00007C64 3B4678                    <1>         cmp     eax, [esi+LD_Clusters]
```

```
4163 00007C67 77E8              <1>       ja     short retn_from_get_free_fat_clusters
4164 00007C69 40                <1>       inc    eax
4165                            <1>       ;mov   [FAT_CurrentCluster], eax
4166 00007C6A EBDA              <1>       jmp    short loc_gfc_loop_get_next_cluster
4167                            <1>
4168                            <1> floppy_drv_init:
4169                            <1>       ; 09/12/2017
4170                            <1>       ; 06/07/2016
4171                            <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4172                            <1>       ; 24/07/2011
4173                            <1>       ; 04/07/2009
4174                            <1>       ; INPUT ->
4175                            <1>       ;      DL = Drive number (0,1)
4176                            <1>       ; OUTPUT ->
4177                            <1>       ;      BL = drive name
4178                            <1>       ;      BH = drive number
4179                            <1>       ;      ESI = Logical DOS drv description table
4180                            <1>       ;      EAX = Volume serial number
4181                            <1>
4182 00007C6C BE[CE6C0000]     <1>       mov    esi, fd0_type ; 10/01/2016
4183 00007C71 BF00010900        <1>       mov    edi, Logical_DOSDisks
4184 00007C76 08D2              <1>       or     dl, dl
4185 00007C78 7407              <1>       jz     short loc_drv_init_fd0_fd1
4186 00007C7A 81C700010000      <1>       add    edi, 100h
4187 00007C80 46                <1>       inc    esi ; fd1_type ; 10/01/2016
4188                            <1> loc_drv_init_fd0_fd1:
4189 00007C81 C6477E00          <1>       mov    byte [edi+LD_MediaChanged], 0
4190 00007C85 803E01            <1>       cmp    byte [esi], 1 ; type (>0 if it is existing)
4191                            <1>              ; 4 = 1.44 MB, 80 track, 3 1/2"
4192 00007C88 7221              <1>       jb     short read_fd_boot_sector_retn
4193 00007C8A 885702            <1>       mov    [edi+LD_PhyDrvNo], dl
4194                            <1> read_fd_boot_sector:
4195 00007C8D 30F6              <1>       xor    dh, dh
4196 00007C8F B904000000        <1>       mov    ecx, 4 ; Retry Count
4197                            <1> read_fd_boot_sector_again:
4198 00007C94 51                <1>       push   ecx
4199                            <1>       ;mov   cx, 1
4200 00007C95 B101              <1>       mov    cl, 1
4201 00007C97 66B80102          <1>       mov    ax, 0201h ; Read 1 sector
4202 00007C9B BB[C68E0100]      <1>       mov    ebx, DOSBootSectorBuff
4203 00007CA0 E8E4D5FFFF        <1>       call   int13h
4204 00007CA5 59                <1>       pop    ecx
4205 00007CA6 7304              <1>       jnc    short use_fd_boot_sector_params
4206 00007CA8 E2EA              <1>       loop   read_fd_boot_sector_again
4207                            <1>
4208                            <1> read_fd_boot_sector_stc_retn:
4209 00007CAA F9                <1>       stc
4210                            <1> read_fd_boot_sector_retn:
4211 00007CAB C3                <1>       retn
4212                            <1>
4213                            <1> use_fd_boot_sector_params:
4214                            <1>       ;mov   esi, DOSBootSectorBuff
4215 00007CAC 89DE              <1>       mov    esi, ebx
4216 00007CAE 6681BEFE01000055AA <1>      cmp    word [esi+BS_Validation], 0AA55h
4217 00007CB7 75F1              <1>       jne    short read_fd_boot_sector_stc_retn
4218 00007CB9 66817E035346      <1>         cmp     word [esi+bs_FS_Identifier], 'SF'
4219 00007CBF 0F85A2000000      <1>         jne     use_fd_fatfs_boot_sector_params
4220                            <1>       ;
4221 00007CC5 8A462D            <1>       mov    al, [esi+bs_FS_LBA_Ready]
4222 00007CC8 884705            <1>       mov    [edi+LD_FS_LBAYes], al
4223                            <1>       ;
4224                            <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
4225 00007CCB 8A4608            <1>       mov    al, [esi+bs_FS_MediaAttrib]
4226 00007CCE 884706            <1>       mov    [edi+LD_FS_MediaAttrib], al
4227                            <1>       ;
4228 00007CD1 8A460A            <1>         mov al, [esi+bs_FS_VersionMaj]
4229 00007CD4 884707            <1>       mov    byte [edi+LD_FS_VersionMajor], al
4230 00007CD7 668B4606          <1>       mov    ax, [esi+bs_FS_BytesPerSec]
4231 00007CDB 66894711          <1>       mov    [edi+LD_FS_BytesPerSec], ax
4232 00007CDF 8A462E            <1>       mov    al, [esi+bs_FS_SecPerTrack]
4233 00007CE2 28E4              <1>       sub    ah, ah ; 09/12/2017
4234 00007CE4 6689471E          <1>       mov    [edi+LD_FS_SecPerTrack], ax
4235 00007CE8 8A462F            <1>       mov    al, [esi+bs_FS_Heads]
4236 00007CEB 66894720          <1>       mov    [edi+LD_FS_NumHeads], ax
4237                            <1>       ;
4238 00007CEF 8B4628            <1>       mov    eax, [esi+bs_FS_UnDelDirD]
4239 00007CF2 894722            <1>       mov    [edi+LD_FS_UnDelDirD], eax
4240 00007CF5 8B4618            <1>       mov    eax, [esi+bs_FS_MATLocation]
4241 00007CF8 89470C            <1>       mov    [edi+LD_FS_MATLocation], eax
4242 00007CFB 8B461C            <1>       mov    eax, [esi+bs_FS_RootDirD]
4243 00007CFE 894708            <1>       mov    [edi+LD_FS_RootDirD], eax
4244 00007D01 8B460C            <1>       mov    eax, [esi+bs_FS_BeginSector]
4245 00007D04 89476C            <1>       mov    [edi+LD_FS_BeginSector], eax
4246 00007D07 8B4610            <1>       mov    eax, [esi+bs_FS_VolumeSize]
4247 00007D0A 894770            <1>       mov    [edi+LD_FS_VolumeSize], eax
4248                            <1>       ;
4249 00007D0D 89FE              <1>       mov    esi, edi
4250 00007D0F 8B460C            <1>       mov    eax, [esi+LD_FS_MATLocation]
4251                            <1>       ;add   eax, [edi+LD_FS_BeginSector]
4252                            <1> read_fd_MAT_sector_again:
4253                            <1>       ;mov   ebx, DOSBootSectorBuff
4254                            <1>       ;mov   ecx, 1
4255 00007D12 B101              <1>       mov    cl, 1
4256 00007D14 E86AAD0000        <1>       call   chs_read
4257 00007D19 89DE              <1>       mov    esi, ebx
4258 00007D1B 7301              <1>       jnc    short use_fdfs_mat_sector_params
4259                            <1>       ;jmp   short read_fd_boot_sector_retn
4260 00007D1D C3                <1>       retn
4261                            <1> use_fdfs_mat_sector_params:
4262 00007D1E 8B460C            <1>       mov    eax, [esi+FS_MAT_DATLocation]
4263 00007D21 894714            <1>       mov    [edi+LD_FS_DATLocation], eax
4264 00007D24 8B4610            <1>       mov    eax, [esi+FS_MAT_DATScount]
4265 00007D27 894718            <1>       mov    [edi+LD_FS_DATSectors], eax
4266 00007D2A 8B4714            <1>       mov    eax, [edi+FS_MAT_FreeSectors]
4267 00007D2D 894774            <1>       mov    [edi+LD_FS_FreeSectors], eax
```

```
4268 00007D30 8B4618              <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
4269 00007D33 894778              <1>       mov    [edi+LD_FS_FirstFreeSector], eax
4270                              <1>       ;
4271 00007D36 89FE                <1>       mov    esi, edi
4272 00007D38 8B4608              <1>       mov    eax, [esi+LD_FS_RootDirD]
4273                              <1> read_fd_RDT_sector_again:
4274                              <1>       ;mov   ebx, DOSBootSectorBuff
4275                              <1>       ;mov   cx, 1
4276 00007D3B B101                <1>       mov    cl, 1
4277 00007D3D E841AD0000          <1>       call   chs_read
4278 00007D42 89DE                <1>       mov    esi, ebx
4279 00007D44 7220                <1>       jc     short read_fd_RDT_sector_retn
4280                              <1> use_fdfs_RDT_sector_params:
4281 00007D46 8B461C              <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
4282 00007D49 894728              <1>       mov    [edi+LD_FS_VolumeSerial], eax
4283 00007D4C 57                  <1>       push   edi
4284                              <1>       ;mov   ecx, 16
4285 00007D4D B110                <1>       mov    cl, 16
4286 00007D4F 83C640              <1>       add    esi, FS_RDT_VolumeName
4287 00007D52 83C72C              <1>       add    edi, LD_FS_VolumeName
4288 00007D55 F3A5                <1>       rep    movsd ; 64 bytes
4289 00007D57 5E                  <1>       pop    esi
4290 00007D58 C6460300            <1>       mov    byte [esi+LD_FATType], 0
4291 00007D5C C64604A1            <1>       mov    byte [esi+LD_FSType], 0A1h
4292 00007D60 E9A5000000          <1>         jmp     loc_cont_use_fd_boot_sector_params
4293                              <1>
4294                              <1> read_fd_RDT_sector_stc_retn:
4295 00007D65 F9                  <1>       stc
4296                              <1> read_fd_RDT_sector_retn:
4297 00007D66 C3                  <1>       retn
4298                              <1>
4299                              <1> use_fd_fatfs_boot_sector_params:
4300 00007D67 807E2629            <1>       cmp    byte [esi+BS_BootSig], 29h
4301 00007D6B 75F8                <1>       jne    short read_fd_RDT_sector_stc_retn
4302 00007D6D 807E15F0            <1>       cmp    byte [esi+BPB_Media], 0F0h
4303 00007D71 72F3                <1>       jb     short read_fd_RDT_sector_retn
4304 00007D73 57                  <1>       push   edi
4305 00007D74 83C706              <1>       add    edi, LD_BPB
4306                              <1>       ;mov   ecx, 16
4307 00007D77 B110                <1>       mov    cl, 16
4308 00007D79 F3A5                <1>       rep    movsd ; 64 bytes
4309 00007D7B 5E                  <1>       pop    esi
4310 00007D7C 31C0                <1>       xor    eax, eax
4311 00007D7E 89466C              <1>       mov    [esi+LD_StartSector], eax ; 0
4312 00007D81 668B461C            <1>       mov    ax, [esi+LD_BPB+BPB_FATSz16]
4313 00007D85 8A4E16              <1>       mov    cl, [esi+LD_BPB+BPB_NumFATs]
4314 00007D88 F7E1                <1>       mul    ecx
4315                              <1>       ; edx = 0 !
4316 00007D8A 668B5614            <1>       mov    dx, [esi+LD_BPB+BPB_RsvdSecCnt]
4317 00007D8E 66895660            <1>       mov    [esi+LD_FATBegin], dx
4318                              <1>       ;add   eax, edx
4319 00007D92 6601D0              <1>       add    ax, dx
4320 00007D95 894664              <1>       mov    [esi+LD_ROOTBegin], eax
4321 00007D98 894668              <1>       mov    [esi+LD_DATABegin], eax
4322 00007D9B 668B5617            <1>       mov    dx, [esi+LD_BPB+BPB_RootEntCnt]
4323                              <1>       ;;shl  edx, 5 ; * 32 (Size of a directory entry)
4324                              <1>       ;shl   dx, 5
4325                              <1>       ;;add  edx, 511
4326                              <1>       ;add   dx, 511
4327                              <1>       ;;shr  edx, 9 ; edx = ((edx*32)+511) / 512
4328                              <1>       ;shr   dx, 9
4329 00007D9F 6683C20F            <1>       add    dx, 15 ; 06/07/2016 (+(512/32)-1)
4330 00007DA3 66C1EA04            <1>       shr    dx, 4 ; / 16 (==16 entries per sector)
4331 00007DA7 015668              <1>       add    [esi+LD_DATABegin], edx ; + rd sectors
4332                              <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
4333 00007DAA 668B4619            <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
4334 00007DAE 894670              <1>       mov    [esi+LD_TotalSectors], eax
4335 00007DB1 2B4668              <1>       sub    eax, [esi+LD_DATABegin]
4336                              <1>       ;movzx ecx, byte [esi+LD_BPB+BPB_SecPerClust]
4337 00007DB4 8A4E13              <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
4338 00007DB7 80F901              <1>       cmp    cl, 1
4339 00007DBA 7605                <1>       jna    short save_fd_fatfs_cluster_count
4340                              <1>       ;sub   edx, edx
4341 00007DBC 6629D2              <1>       sub    dx, dx ; 0
4342                              <1>       ;sub   dl, dl ; 06/07/2016
4343 00007DBF F7F1                <1>       div    ecx
4344                              <1> save_fd_fatfs_cluster_count:
4345 00007DC1 894678              <1>       mov    [esi+LD_Clusters], eax
4346                              <1>
4347                              <1>       ; Maximum Valid Cluster Number = EAX +1
4348                              <1>       ; with 2 reserved clusters= EAX +2
4349                              <1>
4350                              <1> reset_FAT_buffer_decriptors:
4351 00007DC4 29C0                <1>       sub    eax, eax ; 0
4352 00007DC6 A2[CA900100]        <1>       mov    [FAT_BuffValidData], al ; 0
4353 00007DCB A2[CB900100]        <1>       mov    [FAT_BuffDrvName], al ; 0
4354 00007DD0 A3[CE900100]        <1>       mov    [FAT_BuffSector], eax ; 0
4355                              <1>
4356                              <1> read_fd_FAT_sectors:
4357 00007DD5 BB001C0900          <1>       mov    ebx, FAT_Buffer
4358 00007DDA 668B4614            <1>       mov    ax, [esi+LD_BPB+BPB_RsvdSecCnt]
4359                              <1>       ;mov   ecx, 3
4360 00007DDE B103                <1>       mov    cl, 3 ; 3 sectors
4361 00007DE0 E89EAC0000          <1>       call   chs_read
4362 00007DE5 7240                <1>       jc     short read_fd_FAT_sectors_retn
4363                              <1> use_fd_FAT_sectors:
4364 00007DE7 8A4602              <1>       mov    al, [esi+LD_PhyDrvNo]
4365 00007DEA 0441                <1>       add    al, 'A'
4366 00007DEC A2[CB900100]        <1>       mov    [FAT_BuffDrvName], al
4367 00007DF1 C605[CA900100]01    <1>       mov    byte [FAT_BuffValidData], 1
4368 00007DF8 E82B000000          <1>       call   fd_init_calculate_free_clusters
4369 00007DFD 7228                <1>       jc     short read_fd_FAT_sectors_retn
4370                              <1>
4371                              <1> loc_use_fd_boot_sector_params_FAT:
4372 00007DFF C6460301            <1>       mov    byte [esi+LD_FATType], 1 ; FAT 12
```

```
4373 00007E03 C6460401            <1>       mov    byte [esi+LD_FSType], 1
4374 00007E07 8B462D              <1>         mov     eax, [esi+LD_BPB+VolumeID]
4375                              <1> loc_cont_use_fd_boot_sector_params:
4376 00007E0A 8A7E02              <1>       mov    bh, [esi+LD_PhyDrvNo]
4377 00007E0D 887E7D              <1>       mov    [esi+LD_DParamEntry], bh
4378 00007E10 88FB                <1>       mov    bl, bh
4379 00007E12 80C341              <1>       add    bl, 'A'
4380 00007E15 881E                <1>       mov    byte [esi+LD_Name], bl
4381 00007E17 C6460101            <1>       mov    byte [esi+LD_DiskType], 1
4382 00007E1B C6460500            <1>       mov    byte [esi+LD_LBAYes], 0
4383 00007E1F C6467C00            <1>       mov    byte [esi+LD_PartitionEntry], 0
4384 00007E23 C6467E06            <1>       mov    byte [esi+LD_MediaChanged], 6 ; Volume Name Reset
4385                              <1>
4386                              <1> read_fd_FAT_sectors_retn:
4387 00007E27 C3                  <1>       retn
4388                              <1>
4389                              <1> fd_init_calculate_free_clusters:
4390                              <1>       ; 09/12/2017
4391                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4392                              <1>       ; 04/07/2009
4393                              <1>       ; INPUT ->
4394                              <1>       ;     ESI = Logical DOS drive description table address
4395                              <1>       ; OUTPUT ->
4396                              <1>       ;    [ESI+LD_FreeSectors] will be set
4397                              <1>
4398 00007E28 29C0                <1>       sub    eax, eax
4399 00007E2A 894674              <1>       mov    [esi+LD_FreeSectors], eax ; 0
4400 00007E2D B002                <1>       mov    al, 2 ; eax = 2
4401                              <1>
4402                              <1> fd_init_loop_get_next_cluster:
4403 00007E2F E830000000          <1>       call   fd_init_get_next_cluster
4404 00007E34 722D                <1>       jc     short fd_init_calculate_free_clusters_retn
4405                              <1>
4406                              <1> fd_init_free_fat_clusters:
4407                              <1>       ;cmp   eax, 0
4408                              <1>       ;ja    short fd_init_pass_inc_free_cluster_count
4409                              <1>       ;and   eax, eax
4410                              <1>       ;jnz   short fd_init_pass_inc_free_cluster_count
4411 00007E36 6621C0              <1>       and    ax, ax
4412 00007E39 7504                <1>       jnz    short fd_init_pass_inc_free_cluster_count
4413                              <1>       ;inc   dword [esi+LD_FreeSectors]
4414 00007E3B 66FF4674            <1>         inc  word [esi+LD_FreeSectors]
4415                              <1>
4416                              <1> fd_init_pass_inc_free_cluster_count:
4417                              <1>       ;mov   eax, [FAT_CurrentCluster]
4418 00007E3F 66A1[C6900100]      <1>       mov    ax, [FAT_CurrentCluster]
4419                              <1>       ;cmp   eax, [esi+LD_Clusters]
4420 00007E45 663B4678            <1>       cmp    ax, [esi+LD_Clusters]
4421 00007E49 7704                <1>       ja     short short retn_from_fd_init_calculate_free_clusters
4422                              <1>       ;inc   eax
4423 00007E4B 6640                <1>       inc    ax
4424 00007E4D EBE0                <1>       jmp    short fd_init_loop_get_next_cluster
4425                              <1>
4426                              <1> retn_from_fd_init_calculate_free_clusters:
4427 00007E4F 8A4613              <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust]
4428 00007E52 3C01                <1>       cmp    al, 1
4429 00007E54 760D                <1>       jna    short fd_init_calculate_free_clusters_retn
4430                              <1>       ;movzx eax, al
4431 00007E56 30E4                <1>       xor    ah, ah ; 09/12/2017
4432                              <1>       ;mov   ecx, [esi+LD_FreeSectors]
4433 00007E58 668B4E74            <1>       mov    cx, [esi+LD_FreeSectors] ; Count of free clusters
4434                              <1>       ;mul   ecx
4435 00007E5C 66F7E1              <1>       mul    cx
4436                              <1>       ;mov   [esi+LD_FreeSectors], eax
4437 00007E5F 66894674            <1>       mov    [esi+LD_FreeSectors], ax
4438                              <1> fd_init_calculate_free_clusters_retn:
4439 00007E63 C3                  <1>       retn
4440                              <1>
4441                              <1> fd_init_get_next_cluster:
4442                              <1>       ; 04/02/2016
4443                              <1>       ; 02/02/2016
4444                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4445                              <1>       ; 04/07/2009
4446                              <1>       ; INPUT ->
4447                              <1>       ;    EAX = Current cluster
4448                              <1>       ;    ESI = Logical DOS drive description table address
4449                              <1>       ;    EDX = 0
4450                              <1>       ; OUTPUT ->
4451                              <1>       ;    EAX = Next cluster
4452                              <1>
4453 00007E64 A3[C6900100]        <1>       mov    [FAT_CurrentCluster], eax
4454                              <1> fd_init_get_next_cluster_readnext:
4455 00007E69 29D2                <1>       sub    edx, edx ; 0
4456 00007E6B BB00040000          <1>       mov    ebx, 1024 ; 400h
4457 00007E70 F7F3                <1>       div    ebx
4458                              <1>       ; EAX = Count of 3 FAT sectors
4459                              <1>       ; EDX = Buffer entry index
4460 00007E72 89C1                <1>       mov    ecx, eax
4461                              <1>       ;mov   eax, 3
4462 00007E74 B003                <1>       mov    al, 3
4463 00007E76 F7E2                <1>       mul    edx ; Multiply by 3
4464 00007E78 66D1E8              <1>       shr    ax, 1 ; Divide by 2
4465 00007E7B 89C3                <1>       mov    ebx, eax ; Buffer byte offset
4466 00007E7D 81C3001C0900        <1>       add    ebx, FAT_Buffer
4467 00007E83 89C8                <1>       mov    eax, ecx
4468                              <1>       ;mov   edx, 3
4469 00007E85 66BA0300            <1>       mov    dx, 3
4470 00007E89 F7E2                <1>       mul    edx
4471                              <1>       ; EAX = FAT Beginning Sector
4472                              <1>       ; EDX = 0
4473 00007E8B 8A0E                <1>       mov    cl, [esi+LD_Name]
4474                              <1>       ;cmp   byte [FAT_BuffValidData], 0
4475                              <1>       ;jna   short fd_init_load_FAT_sectors0
4476 00007E8D 3A0D[CB900100]      <1>       cmp    cl, [FAT_BuffDrvName]
4477 00007E93 751E                <1>       jne    short fd_init_load_FAT_sectors0
```

```
4478 00007E95 3B05[CE900100]      <1>       cmp    eax, [FAT_BuffSector]
4479 00007E9B 751C                <1>       jne    short fd_init_load_FAT_sectors1
4480                              <1>       ;mov   eax, [FAT_CurrentCluster]
4481 00007E9D A0[C6900100]        <1>       mov    al, [FAT_CurrentCluster]
4482                              <1>       ;shr   eax, 1
4483 00007EA2 D0E8                <1>       shr    al, 1
4484 00007EA4 668B03              <1>       mov    ax, [ebx]
4485 00007EA7 7306                <1>       jnc    short fd_init_gnc_even
4486 00007EA9 66C1E804            <1>       shr    ax, 4
4487                              <1> fd_init_gnc_clc_retn:
4488 00007EAD F8                  <1>       clc
4489 00007EAE C3                  <1>       retn
4490                              <1>
4491                              <1> fd_init_gnc_even:
4492 00007EAF 80E40F              <1>       and    ah, 0Fh
4493 00007EB2 C3                  <1>       retn
4494                              <1>
4495                              <1> fd_init_load_FAT_sectors0:
4496 00007EB3 880D[CB900100]      <1>       mov    [FAT_BuffDrvName], cl
4497                              <1> fd_init_load_FAT_sectors1:
4498 00007EB9 C605[CA900100]00    <1>       mov    byte [FAT_BuffValidData], 0
4499 00007EC0 A3[CE900100]        <1>       mov    [FAT_BuffSector], eax
4500 00007EC5 034660              <1>       add    eax, [esi+LD_FATBegin]
4501 00007EC8 BB001C0900          <1>       mov    ebx, FAT_Buffer
4502                              <1>       ;movzx ecx, word [esi+LD_BPB+BPB_FATSz16]
4503 00007ECD 668B4E1C            <1>       mov    cx, [esi+LD_BPB+BPB_FATSz16]
4504 00007ED1 662B0D[CE900100]    <1>       sub    cx, [FAT_BuffSector]
4505                              <1>         ;cmp ecx, 3
4506 00007ED8 6683F903            <1>       cmp    cx, 3
4507 00007EDC 7605                <1>       jna    short fdinit_pass_fix_sector_count_3
4508                              <1>       ;mov   ecx, 3
4509 00007EDE B903000000          <1>       mov    ecx, 3
4510                              <1> fdinit_pass_fix_sector_count_3:
4511 00007EE3 E89BAB0000          <1>       call   chs_read
4512 00007EE8 730D                <1>       jnc    short fd_init_FAT_sectors_no_load_error
4513 00007EEA C605[CA900100]00    <1>       mov    byte [FAT_BuffValidData], 0
4514                              <1>              ; Drv not ready or read Error !
4515 00007EF1 B80F000000          <1>       mov    eax, ERR_DRV_NOT_RDY ; 15
4516                              <1>       ;xor   edx, edx
4517 00007EF6 C3                  <1>       retn
4518                              <1>
4519                              <1> fd_init_FAT_sectors_no_load_error:
4520 00007EF7 C605[CA900100]01    <1>       mov    byte [FAT_BuffValidData], 1
4521 00007EFE A1[C6900100]        <1>       mov    eax, [FAT_CurrentCluster]
4522 00007F03 E961FFFFFF          <1>         jmp     fd_init_get_next_cluster_readnext
4523                              <1>
4524                              <1> get_FAT_volume_name:
4525                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4526                              <1>       ; 12/09/2009
4527                              <1>       ; INPUT ->
4528                              <1>       ;      BH = Logical DOS drive number (0,1,2,3,4 ...)
4529                              <1>       ;       BL = 0
4530                              <1>       ; OUTPUT ->
4531                              <1>       ;      CF = 0 -> ESI = Volume name address
4532                              <1>       ;      CF = 1 -> Root volume name not found
4533                              <1>
4534                              <1>       ;mov   ah, 0FFh
4535                              <1>       ;mov   al, [Last_Dos_DiskNo]
4536                              <1>       ;cmp   al, bh
4537                              <1>       ;jb     short loc_gfvn_dir_load_err
4538                              <1>
4539 00007F08 89DE                <1>       mov    esi, ebx
4540 00007F0A 81E600FF0000        <1>       and    esi, 0FF00h ; esi = bh
4541 00007F10 81C600010900        <1>       add    esi, Logical_DOSDisks
4542 00007F16 8A06                <1>       mov     al, [esi+LD_Name]
4543 00007F18 8A6603              <1>       mov     ah, [esi+LD_FATType]
4544 00007F1B 80FC01              <1>       cmp     ah, 1
4545 00007F1E 7210                <1>       jb     short loc_gfvn_dir_load_err
4546 00007F20 3C41                <1>       cmp    al, 'A'
4547 00007F22 720C                <1>       jb      short loc_gfvn_dir_load_err
4548 00007F24 80FC02              <1>       cmp    ah, 2
4549 00007F27 7708                <1>       ja      short get_FAT32_root_cluster
4550                              <1>
4551 00007F29 E8634E0000          <1>       call    load_FAT_root_directory
4552 00007F2E 730B                <1>       jnc     short loc_get_volume_name
4553                              <1>
4554                              <1> loc_gfvn_dir_load_err:
4555 00007F30 C3                  <1>       retn
4556                              <1>
4557                              <1> get_FAT32_root_cluster:
4558 00007F31 8B4632              <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
4559 00007F34 E8E34E0000          <1>       call    load_FAT_sub_directory
4560 00007F39 7224                <1>       jc     short loc_get_volume_name_retn
4561                              <1>
4562                              <1> loc_get_volume_name:
4563 00007F3B BE00000800          <1>         mov     esi, Directory_Buffer
4564 00007F40 6631C9              <1>       xor    cx, cx ; 0
4565                              <1> check_root_volume_name:
4566 00007F43 8A06                <1>       mov    al, [esi]
4567 00007F45 08C0                <1>       or      al, al
4568 00007F47 7416                <1>       jz      short loc_get_volume_name_retn
4569 00007F49 807E0B08            <1>       cmp     byte [esi+0Bh], 08h
4570 00007F4D 7410                <1>       je      short loc_get_volume_name_retn
4571 00007F4F 663B0D[DF900100]    <1>       cmp     cx, [DirBuff_LastEntry]
4572 00007F56 7308                <1>       jnb     short pass_check_root_volume_name
4573 00007F58 6641                <1>       inc     cx
4574 00007F5A 83C620              <1>       add     esi, 32
4575 00007F5D EBE4                <1>       jmp     short check_root_volume_name
4576                              <1>
4577                              <1> loc_get_volume_name_retn:
4578 00007F5F C3                  <1>       retn
4579                              <1>
4580                              <1> pass_check_root_volume_name:
4581 00007F60 803D[DB900100]03    <1>       cmp    byte [DirBuff_FATType], 3
4582 00007F67 7230                <1>       jb     short loc_get_volume_name_retn_xor
```

```
4583                                    <1>
4584 00007F69 BB001C0900                <1>       mov    ebx, FAT_Buffer
4585 00007F6E BE00010900                <1>       mov    esi, Logical_DOSDisks
4586 00007F73 31C0                      <1>       xor    eax, eax
4587 00007F75 8A25[DA900100]            <1>       mov    ah, [DirBuff_DRV]
4588 00007F7B 80EC41                    <1>       sub    ah, 'A'
4589 00007F7E 01C6                      <1>       add    esi, eax
4590 00007F80 A1[E1900100]              <1>       mov    eax, [DirBuff_Cluster]
4591 00007F85 E8AC4C0000                <1>       call   get_next_cluster
4592 00007F8A 7305                      <1>       jnc    short loc_gfvn_load_FAT32_dir_cluster
4593                                    <1>
4594 00007F8C 83F801                    <1>       cmp     eax, 1
4595 00007F8F F5                        <1>       cmc
4596 00007F90 C3                        <1>       retn
4597                                    <1>
4598                                    <1> loc_gfvn_load_FAT32_dir_cluster:
4599 00007F91 E8864E0000                <1>       call   load_FAT_sub_directory
4600 00007F96 73A3                      <1>       jnc    short loc_get_volume_name
4601 00007F98 C3                        <1>       retn
4602                                    <1>
4603                                    <1> loc_get_volume_name_retn_xor:
4604 00007F99 31C0                      <1>       xor    eax, eax
4605 00007F9B C3                        <1>       retn
4606                                    <1>
4607                                    <1> get_media_change_status:
4608                                    <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4609                                    <1>       ; 09/09/2009
4610                                    <1>       ; INPUT:
4611                                    <1>       ;     DL = Drive number (physical)
4612                                    <1>       ; OUTPUT: clc & AH = 6 media changed
4613                                    <1>       ;     clc & AH = 0 media not changed
4614                                    <1>       ;     stc -> Drive not ready or an error
4615                                    <1>
4616 00007F9C B416                      <1>       mov    ah, 16h
4617 00007F9E E8E6D2FFFF                <1>       call   int13h
4618 00007FA3 80FC06                    <1>       cmp    ah, 06h
4619 00007FA6 7405                      <1>       je     short loc_gmc_status_retn
4620 00007FA8 08E4                      <1>       or     ah, ah
4621 00007FAA 7401                      <1>       jz     short loc_gmc_status_retn
4622                                    <1> loc_gmc_status_stc_retn:
4623 00007FAC F9                        <1>       stc
4624                                    <1> loc_gmc_status_retn:
4625 00007FAD C3                        <1>       retn
3086                                        %include 'trdosk3.s' ; 06/01/2016
3087                                    <1> ; ****************************************************************************
3088                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk3.s
3089                                    <1> ; ----------------------------------------------------------------------------
3090                                    <1> ; Last Update: 31/12/2017
3091                                    <1> ; ----------------------------------------------------------------------------
3092                                    <1> ; Beginning: 06/01/2016
3093                                    <1> ; ----------------------------------------------------------------------------
3094                                    <1> ; Assembler: NASM version 2.11 (trdos386.s)
3095                                    <1> ; ----------------------------------------------------------------------------
3096                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3097                                    <1> ; MAINPROG.ASM (09/11/2011)
3098                                    <1> ; ****************************************************************************
3099                                    <1> ; MAINPROG.ASM [ TRDOS KERNEL - COMMAND EXECUTER SECTION - MAIN PROGRAM ]
3100                                    <1> ; (c) 2004-2011  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/11/2011
3101                                    <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ] Last Update: 09/11/2011
3102                                    <1> ; DIR.ASM [ DIRECTORY FUNCTIONS ] Last Update: 09/10/2011
3103                                    <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
3104                                    <1>
3105                                    <1> change_current_drive:
3106                                    <1>       ; 16/10/2016
3107                                    <1>       ; 02/02/2016
3108                                    <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
3109                                    <1>       ; 18/08/2011
3110                                    <1>       ; 09/09/2009
3111                                    <1>       ; INPUT:
3112                                    <1>       ;   DL = Logical DOS Drive Number
3113                                    <1>       ; OUTPUT:
3114                                    <1>       ;  cf=1 -> Not successful
3115                                    <1>       ;   EAX = Error code
3116                                    <1>       ;  cf=0 ->
3117                                    <1>       ;   EAX = 0 (successful)
3118                                    <1>
3119 00007FAE 31DB                      <1>       xor    ebx, ebx
3120 00007FB0 88D7                      <1>       mov    bh, dl
3121                                    <1>
3122                                    <1>       ;cmp   dl, 1
3123                                    <1>       ;jna   short loc_ccdrv_initial_media_change_check
3124                                    <1>       ;cmp   bh, [Last_Dos_DiskNo]
3125                                    <1>       ;ja    short loc_ccdrv_drive_not_ready_err
3126                                    <1>
3127                                    <1> loc_ccdrv_initial_media_change_check:
3128 00007FB2 BE00010900                <1>       mov    esi, Logical_DOSDisks
3129 00007FB7 01DE                      <1>       add    esi, ebx
3130                                    <1> loc_ccdrv_dos_drive_name_check:
3131 00007FB9 80FA02                    <1>       cmp    dl, 2
3132 00007FBC 720F                      <1>       jb     short loc_ccdrv_dos_drive_name_check_ok
3133                                    <1>
3134 00007FBE 8A06                      <1>       mov    al, [esi+LD_Name]
3135 00007FC0 2C41                      <1>       sub    al, 'A'
3136 00007FC2 38D0                      <1>       cmp    al, dl
3137 00007FC4 7407                      <1>       je     short loc_ccdrv_dos_drive_name_check_ok
3138                                    <1>
3139                                    <1> loc_ccdrv_drive_not_ready_err:
3140                                    <1>       ; 16/10/2016 (15h -> 15)
3141 00007FC6 B80F000000                <1>       mov    eax, 15 ; Drive not ready
3142                                    <1> loc_change_current_drive_stc_retn:
3143 00007FCB F9                        <1>       stc
3144 00007FCC C3                        <1>       retn
3145                                    <1>
3146                                    <1> loc_ccdrv_dos_drive_name_check_ok:
3147 00007FCD 8A667E                    <1>       mov    ah, [esi+LD_MediaChanged]
```

```
3148 00007FD0 80FC06              <1>       cmp    ah, 6  ; VOLUME NAME CHECK/MOVE SIGN
3149 00007FD3 7455                <1>       je     short loc_ccdrv_get_FAT_volume_name_0
3150                              <1>
3151 00007FD5 80FA01              <1>       cmp    dl, 1
3152 00007FD8 777D                <1>       ja     short loc_gmcs_init_drv_hd
3153                              <1>
3154                              <1> loc_gmcs_init_drv_fd:
3155 00007FDA 08E4                <1>       or     ah, ah
3156                              <1>       ; AH = 1 is initialization sign (invalid_fd_parameter)
3157 00007FDC 7517                <1>       jnz    short loc_ccdrv_call_fd_init
3158                              <1>
3159 00007FDE E8B9FFFFFF          <1>       call   get_media_change_status
3160 00007FE3 72E1                <1>       jc     short loc_ccdrv_drive_not_ready_err
3161                              <1>
3162 00007FE5 20E4                <1>       and    ah, ah
3163 00007FE7 7476                <1>       jz     short loc_change_current_drv3
3164                              <1>
3165 00007FE9 80F406              <1>       xor    ah, 6
3166 00007FEC 75D8                <1>       jnz    short loc_ccdrv_drive_not_ready_err
3167                              <1>
3168                              <1> loc_ccdrv_call_fd_init_check_vol_id:
3169 00007FEE E8440A0000          <1>       call   get_volume_serial_number
3170 00007FF3 730D                <1>       jnc    short loc_ccdrv_check_vol_serial
3171                              <1>
3172                              <1> loc_ccdrv_call_fd_init:
3173 00007FF5 E872FCFFFF          <1>       call   floppy_drv_init
3174 00007FFA 731A                <1>       jnc    short loc_reset_drv_fd_current_dir
3175                              <1>
3176                              <1> loc_ccdrv_fdinit_fail_retn:
3177                              <1>       ; 16/10/2016
3178 00007FFC B80F000000          <1>       mov    eax, 15 ; Drive not ready
3179 00008001 C3                  <1>       retn
3180                              <1>
3181                              <1> loc_ccdrv_check_vol_serial:
3182 00008002 A3[AC890100]        <1>       mov    [Current_VolSerial], eax
3183                              <1>       ;mov   dl, bh
3184 00008007 E860FCFFFF          <1>       call   floppy_drv_init
3185 0000800C 72EE                <1>       jc     short loc_ccdrv_fdinit_fail_retn
3186                              <1>
3187 0000800E 3B05[AC890100]      <1>       cmp    eax, [Current_VolSerial]
3188 00008014 7445                <1>       je     short loc_change_current_drv2
3189                              <1>
3190                              <1> loc_reset_drv_fd_current_dir:
3191 00008016 31C0                <1>       xor    eax, eax
3192 00008018 88467F              <1>         mov [esi+LD_CDirLevel], al
3193 0000801B 89F7                <1>       mov    edi, esi
3194 0000801D 81C780000000        <1>       add    edi, LD_CurrentDirectory
3195 00008023 B920000000          <1>       mov    ecx, 32
3196 00008028 F3AB                <1>       rep    stosd
3197                              <1>
3198                              <1> loc_ccdrv_get_FAT_volume_name_0:
3199 0000802A 8A4603              <1>       mov    al, [esi+LD_FATType]
3200 0000802D 08C0                <1>       or     al, al
3201 0000802F 742A                <1>       jz     short loc_change_current_drv2
3202                              <1>
3203 00008031 56                  <1>       push   esi
3204 00008032 3C02                <1>       cmp    al, 2
3205 00008034 7705                <1>       ja     short loc_ccdrv_get_FAT32_vol_name
3206                              <1>
3207                              <1> loc_ccdrv_get_FAT2_16_vol_name:
3208 00008036 83C631              <1>       add    esi, LD_BPB + VolumeLabel
3209 00008039 EB03                <1>       jmp    short loc_ccdrv_get_FAT_volume_name_1
3210                              <1>
3211                              <1> loc_ccdrv_get_FAT32_vol_name:
3212 0000803B 83C64D              <1>       add    esi, LD_BPB + FAT32_VolLab
3213                              <1> loc_ccdrv_get_FAT_volume_name_1:
3214 0000803E 53                  <1>       push   ebx
3215 0000803F 56                  <1>       push   esi
3216 00008040 E8C3FEFFFF          <1>       call   get_FAT_volume_name
3217 00008045 5F                  <1>       pop    edi
3218 00008046 5B                  <1>       pop    ebx
3219                              <1>       ; BL = 0
3220 00008047 720B                <1>       jc     short loc_change_current_drv1
3221 00008049 20C0                <1>       and    al, al
3222 0000804B 7407                <1>       jz     short loc_change_current_drv1
3223                              <1>
3224                              <1> loc_ccdrv_move_FAT_volume_name:
3225 0000804D B90B000000          <1>       mov    ecx, 11
3226 00008052 F3A4                <1>       rep    movsb
3227                              <1>
3228                              <1> loc_change_current_drv1:
3229 00008054 5E                  <1>       pop    esi
3230 00008055 EB04                <1>       jmp    short loc_change_current_drv2
3231                              <1>
3232                              <1> loc_gmcs_init_drv_hd:
3233 00008057 08E4                <1>       or     ah, ah
3234 00008059 7404                <1>       jz     short loc_change_current_drv3
3235                              <1>       ; BL = 0, BH = Logical DOS drive number
3236                              <1> loc_change_current_drv2:
3237 0000805B C6467E00            <1>       mov    byte [esi+LD_MediaChanged], 0
3238                              <1> loc_change_current_drv3:
3239 0000805F 883D[B6890100]      <1>       mov    [Current_Drv], bh
3240                              <1>
3241                              <1>       ;call  restore_current_directory
3242                              <1>       ;retn
3243                              <1>
3244                              <1> restore_current_directory:
3245                              <1>       ; 11/02/2016
3246                              <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
3247                              <1>       ; 25/01/2010
3248                              <1>       ; 12/10/2009
3249                              <1>       ;
3250                              <1>       ; INPUT:
3251                              <1>       ;   ESI = Logical DOS Drive Description Table
3252                              <1>       ;
```

```
                        <1>       ; OUTPUT:
                        <1>       ;   ESI = Logical DOS Drive Description Table
                        <1>       ;   EDI = offset Current_Dir_Drv
                        <1>
00008065 8A4603         <1>       mov   al, [esi+LD_FATType]
00008068 A2[B5890100]   <1>       mov   [Current_FATType], al
                        <1>
0000806D 8A26           <1>       mov   ah, [esi+LD_Name]
0000806F 8825[B7890100] <1>       mov   [Current_Dir_Drv], ah
                        <1>
00008075 20C0           <1>       and   al, al
00008077 741D           <1>       jz    short loc_restore_FS_current_directory
                        <1>
                        <1> loc_restore_FAT_current_directory:
00008079 8A667F         <1>       mov   ah, [esi+LD_CDirLevel]
0000807C 8825[B4890100] <1>       mov   [Current_Dir_Level], ah
00008082 08E4           <1>       or    ah, ah
00008084 7416           <1>         jz  short loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster
                        <1>
00008086 0FB6D4         <1>       movzx edx, ah
00008089 C0E204         <1>       shl   dl, 4 ; * 16
0000808C 01F2           <1>         add edx, esi
0000808E 8B828C000000   <1>       mov   eax, [edx+LD_CurrentDirectory+12]
00008094 EB2C           <1>       jmp   short loc_ccdrv_reset_cdir_FAT_fcluster
                        <1>
                        <1> loc_restore_FS_current_directory:
00008096 E8BC4D0000     <1>       call  load_current_FS_directory
0000809B C3             <1>       retn
                        <1>
                        <1> loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster:
0000809C 3C03           <1>       cmp   al, 3
0000809E 7205           <1>       jb    short loc_ccdrv_reset_cdir_FAT_12_16_fcluster
                        <1> loc_ccdrv_reset_cdir_FAT32_fcluster:
000080A0 8B4632         <1>       mov   eax, [esi+LD_BPB+FAT32_RootFClust]
000080A3 EB04           <1>       jmp   short loc_ccdrv_check_rootdir_sign
                        <1> loc_ccdrv_reset_cdir_FAT_12_16_fcluster:
000080A5 30C0           <1>       xor   al, al  ; xor eax, eax
000080A7 31D2           <1>       xor   edx, edx
                        <1> loc_ccdrv_check_rootdir_sign:
000080A9 80BE8000000000 <1>       cmp   byte [esi+LD_CurrentDirectory], 0
000080B0 7510           <1>       jne   short loc_ccdrv_reset_cdir_FAT_fcluster
                        <1> loc_ccdrv_set_rootdir_FAT_fcluster:
000080B2 89868C000000   <1>         mov     [esi+LD_CurrentDirectory+12], eax
000080B8 C78680000000524F4F- <1>  mov   dword [esi+LD_CurrentDirectory], 'ROOT'
000080C1 54             <1>
                        <1>
                        <1> loc_ccdrv_reset_cdir_FAT_fcluster:
000080C2 A3[B0890100]   <1>       mov   [Current_Dir_FCluster], eax
                        <1>
000080C7 BF[13910100]   <1>       mov   edi, PATH_Array
000080CC 89F2           <1>       mov   edx, esi
000080CE 81C680000000   <1>       add   esi, LD_CurrentDirectory
000080D4 B920000000     <1>       mov   ecx, 32
000080D9 F3A5           <1>       rep   movsd
                        <1>
000080DB E84C2D0000     <1>       call  change_prompt_dir_string
                        <1>
000080E0 89D6           <1>       mov   esi, edx
                        <1>
000080E2 29C0           <1>         sub eax, eax
                        <1>        ;sub edx, edx
000080E4 BF[B7890100]   <1>       mov   edi, Current_Dir_Drv
                        <1>
000080E9 A2[833F0100]   <1>       mov   [Restore_CDIR], al ; 0
000080EE 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
000080EF C705[70950100]- <1>      mov   dword [mainprog_return_addr], return_from_cmd_interpreter
000080F5 [A3810000]     <1>
                        <1>
                        <1> loc_TRDOS_prompt:
000080F9 BF[B68A0100]   <1>       mov   edi, TextBuffer
000080FE C6075B         <1>       mov   byte [edi], "["
00008101 47             <1>       inc   edi
00008102 BE[D63F0100]   <1>       mov   esi, TRDOSPromptLabel
                        <1> get_next_prompt_label_char:
00008107 803E20         <1>       cmp   byte [esi], 20h
0000810A 7203           <1>       jb    short pass_prompt_label
0000810C A4             <1>       movsb
0000810D EBF8           <1>       jmp   short get_next_prompt_label_char
                        <1> pass_prompt_label:
0000810F C6075D         <1>       mov   byte [edi], "]"
00008112 47             <1>       inc   edi
00008113 C60720         <1>       mov   byte [edi], 20h
00008116 47             <1>       inc   edi
00008117 BE[B7890100]   <1>       mov   esi, Current_Dir_Drv
0000811C 66A5           <1>       movsw
0000811E A4             <1>       movsb
                        <1> loc_prompt_current_directory:
0000811F 803E20         <1>       cmp   byte [esi], 20h
00008122 7203           <1>       jb    short pass_prompt_current_directory
00008124 A4             <1>       movsb
00008125 EBF8           <1>       jmp   short loc_prompt_current_directory
                        <1> pass_prompt_current_directory:
00008127 C6073E         <1>       mov   byte [edi], '>'
0000812A 47             <1>       inc   edi
```

```
3356 0000812B C60700              <1>       mov   byte [edi], 0
3357 0000812E BE[B68A0100]        <1>       mov   esi, TextBuffer
3358 00008133 E8EDF2FFFF          <1>       call  print_msg
3359                              <1>
3360                              <1>       ;sub  bh, bh ; video page = 0
3361                              <1>       ;call get_cpos ; get cursor position
3362 00008138 668B15[0E890100]    <1>       mov   dx, [CURSOR_POSN] ; video page 0
3363 0000813F 8815[168A0100]      <1>       mov   [CursorColumn], dl
3364                              <1>
3365                              <1>       ; 30/01/2016 (to show cursor on the row, again)
3366                              <1>       ; (Initial color attributes of video page 0 is 0)
3367                              <1>       ; (see: 'StartPMP' in trdos386.s)
3368                              <1>       ;
3369                              <1>       ;mov  edi, 0B8000h ; start of video page 0
3370                              <1>       ;movzx ecx, dl ; column
3371                              <1>       ;mov  al, 80
3372                              <1>       ;mul  dh
3373                              <1>       ;add  ax, cx
3374                              <1>       ;shl  ax, 1 ; character + attribute
3375                              <1>       ;add  di, ax ; (2*80*row) + (2*column)
3376                              <1>       ;neg  cl
3377                              <1>       ;add  cl, 80
3378                              <1>       ;mov  ax, 700h ;  ah = 7 (color attribute)
3379                              <1>       ;rep  stosw
3380                              <1>
3381                              <1> loc_rw_char:
3382 00008145 E899000000          <1>       call  rw_char
3383                              <1> loc_move_command:
3384 0000814A BE[668A0100]        <1>       mov   esi, CommandBuffer
3385 0000814F 89F7                <1>       mov   edi, esi
3386 00008151 31C9                <1>       xor   ecx, ecx
3387                              <1> first_command_char:
3388 00008153 AC                  <1>       lodsb
3389 00008154 3C20                <1>       cmp   al, 20h
3390 00008156 772E                <1>       ja    short pass_space_control
3391 00008158 7241                <1>       jb    short loc_move_cmd_arguments_ok
3392 0000815A 81FE[B58A0100]      <1>       cmp   esi, CommandBuffer + 79
3393 00008160 72F1                <1>       jb    short first_command_char
3394 00008162 EB37                <1>       jmp   short loc_move_cmd_arguments_ok
3395                              <1>
3396                              <1> next_command_char:
3397 00008164 AC                  <1>       lodsb
3398 00008165 3C20                <1>       cmp   al, 20h
3399 00008167 771D                <1>       ja    short pass_space_control
3400 00008169 7230                <1>       jb    short loc_move_cmd_arguments_ok
3401                              <1>
3402                              <1> loc_1st_cmd_arg: ; 30/01/2016
3403 0000816B AC                  <1>       lodsb
3404 0000816C 3C20                <1>       cmp   al, 20h
3405 0000816E 74FB                <1>       je    short loc_1st_cmd_arg
3406 00008170 7229                <1>       jb    short loc_move_cmd_arguments_ok
3407                              <1>
3408 00008172 C60700              <1>         mov     byte [edi], 0
3409 00008175 47                  <1>       inc   edi
3410                              <1>
3411                              <1> loc_move_cmd_arguments:
3412 00008176 AA                  <1>       stosb
3413 00008177 81FE[B58A0100]      <1>       cmp   esi, CommandBuffer + 79
3414 0000817D 731C                <1>       jnb   short loc_move_cmd_arguments_ok
3415 0000817F AC                  <1>         lodsb
3416 00008180 3C20                <1>       cmp   al, 20h
3417 00008182 73F2                <1>       jnb   short loc_move_cmd_arguments
3418 00008184 EB15                <1>       jmp   short loc_move_cmd_arguments_ok
3419                              <1>
3420                              <1> pass_space_control:
3421 00008186 3C61                <1>       cmp   al, 61h
3422 00008188 7206                <1>       jb    short pass_capitalize
3423 0000818A 3C7A                <1>       cmp   al, 7Ah
3424 0000818C 7702                <1>       ja    short pass_capitalize
3425 0000818E 24DF                <1>       and   al, 0DFh
3426                              <1> pass_capitalize:
3427 00008190 AA                  <1>       stosb
3428 00008191 FEC1                <1>       inc     cl
3429 00008193 81FE[B58A0100]      <1>         cmp     esi, CommandBuffer + 79
3430 00008199 72C9                <1>       jb      short next_command_char
3431                              <1>
3432                              <1> loc_move_cmd_arguments_ok:
3433 0000819B C60700              <1>         mov     byte [edi], 0
3434                              <1>
3435                              <1> call_command_interpreter:
3436 0000819E E8CF080000          <1>       call    command_interpreter
3437                              <1>
3438                              <1> return_from_cmd_interpreter:
3439 000081A3 B950000000          <1>         mov ecx, 80
3440                              <1>       ;mov  cx, 80
3441 000081A8 BF[668A0100]        <1>       mov   edi, CommandBuffer
3442 000081AD 30C0                <1>       xor   al, al
3443 000081AF F3AA                <1>       rep   stosb
3444                              <1>       ;cmp  byte [Program_Exit], 0
3445                              <1>       ;ja   short loc_terminate_trdos
3446                              <1>
3447                              <1>       ; 16/01/2016
3448 000081B1 803D[9A6E0000]03    <1>       cmp   byte [CRT_MODE], 3 ; 80*25 color
3449 000081B8 741D                <1>       je    short pass_set_txt_mode
3450                              <1>
3451 000081BA E84199FFFF          <1>       call  set_txt_mode ; set vide mode to 03h
3452                              <1>       ; 07/01/2017
3453 000081BF 30C0                <1>       xor   al, al
3454                              <1>
3455                              <1> loc_check_active_page:
3456                              <1>       ;xor  al, al
3457 000081C1 3805[1E890100]      <1>       cmp   [ACTIVE_PAGE], al ; 0
3458 000081C7 0F842CFFFFFF        <1>         je      loc_TRDOS_prompt
3459                              <1>       ; AL = 0 = video page 0
3460 000081CD E8739DFFFF          <1>       call  set_active_page
```

```
3461 000081D2 E922FFFFFF          <1>         jmp     loc_TRDOS_prompt ; infinitive loop
3462                              <1>
3463                              <1> pass_set_txt_mode:
3464 000081D7 BE[1F4C0100]        <1>         mov     esi, nextline
3465 000081DC E844F2FFFF          <1>         call    print_msg
3466 000081E1 EBDE                <1>         jmp     short loc_check_active_page
3467                              <1>
3468                              <1> rw_char:
3469                              <1>         ; 13/05/2016
3470                              <1>         ; 30/01/2016
3471                              <1>         ; 29/01/2016
3472                              <1>         ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
3473                              <1>         ; 2004-2005
3474                              <1>
3475                              <1>         ; DH = cursor row, DL = cursor column
3476                              <1>         ; BH = 0 = video page number (active page)
3477                              <1>
3478                              <1>         ;xor   bh, bh ; 0 = video page 0
3479                              <1>
3480                              <1> readnextchar:
3481 000081E3 30E4                <1>         xor     ah, ah
3482 000081E5 E8F98CFFFF          <1>         call    int16h
3483 000081EA 20C0                <1>         and     al, al
3484 000081EC 7432                <1>         jz      short loc_arrow
3485 000081EE 3CE0                <1>         cmp     al, 0E0h
3486 000081F0 742E                <1>         je      short loc_arrow
3487 000081F2 3C08                <1>         cmp     al, 08h
3488 000081F4 7542                <1>         jne     short char_return
3489                              <1> loc_back:
3490 000081F6 3A15[168A0100]      <1>         cmp     dl, [CursorColumn]
3491 000081FC 76E5                <1>         jna     short readnextchar
3492                              <1> prev_column:
3493 000081FE FECA                <1>         dec     dl
3494                              <1> set_cursor_pos:
3495                              <1>         ;push  dx
3496 00008200 52                  <1>         push    edx ; 29/12/2017
3497                              <1>         ;xor   bh, bh ; 0 = video page 0
3498                              <1>         ; DH = Row, DL = Column
3499 00008201 E814A1FFFF          <1>         call    _set_cpos ; 17/01/2016
3500 00008206 5A                  <1>         pop  edx ; 29/12/2017
3501                              <1>         ;pop   dx
3502                              <1>         ;movzx ebx, dl
3503 00008207 88D3                <1>         mov     bl, dl
3504 00008209 2A1D[168A0100]      <1>         sub     bl, [CursorColumn]
3505 0000820F B020                <1>         mov     al, 20h
3506 00008211 8883[668A0100]      <1>         mov     [CommandBuffer+ebx], al
3507                              <1>         ;sub   bh, bh ; video page 0
3508                              <1>         ;mov   cx, 1
3509 00008217 B307                <1>         mov     bl, 7 ; color attribute
3510 00008219 E8E39FFFFF          <1>         call    _write_c_current ; 17/01/2016
3511                              <1>         ;mov   dx, [CURSOR_POSN]
3512 0000821E EBC3                <1>         jmp     short readnextchar
3513                              <1> loc_arrow:
3514 00008220 80FC4B              <1>         cmp     ah, 4Bh
3515 00008223 74D1                <1>         je      short loc_back
3516 00008225 80FC53              <1>         cmp     ah, 53h
3517 00008228 74CC                <1>         je      short loc_back
3518 0000822A 80FC4D              <1>         cmp     ah, 4Dh
3519 0000822D 75B4                <1>         jne     short readnextchar
3520 0000822F 80FA4F              <1>         cmp     dl, 79
3521 00008232 73AF                <1>         jnb     short readnextchar
3522 00008234 FEC2                <1>         inc     dl
3523 00008236 EBC8                <1>         jmp     short set_cursor_pos
3524                              <1> char_return:
3525 00008238 0FB6DA              <1>         movzx  ebx, dl
3526 0000823B 2A1D[168A0100]      <1>         sub     bl, [CursorColumn]
3527 00008241 3C20                <1>         cmp     al, 20h
3528 00008243 721D                <1>         jb      short loc_escape
3529 00008245 8883[668A0100]      <1>         mov     [CommandBuffer+ebx], al
3530 0000824B 80FA4F              <1>         cmp     dl, 79
3531 0000824E 7393                <1>         jnb     short readnextchar
3532 00008250 66BB0700            <1>         mov     bx, 7 ; color attribute
3533 00008254 E829A0FFFF          <1>         call    _write_tty
3534 00008259 668B15[0E890100]    <1>         mov     dx, [CURSOR_POSN] ; video page 0
3535 00008260 EB81                <1>         jmp     readnextchar
3536                              <1> loc_escape:
3537 00008262 3C1B                <1>         cmp     al, 1Bh
3538 00008264 7418                <1>         je      short rw_char_retn
3539                              <1>         ;
3540 00008266 3C0D                <1>         cmp     al, 0Dh ; CR
3541 00008268 0F8575FFFFFF        <1>         jne     readnextchar
3542                              <1>         ; 13/05/2016
3543 0000826E 66BB0700            <1>         mov     bx, 7 ; attribute/color (bl)
3544                              <1>                       ; video page 0 (bh=0)
3545 00008272 E80BA0FFFF          <1>         call    _write_tty
3546                              <1>         ;mov   bx, 7  ; attribute/color
3547                              <1>                       ; video page 0 (bh=0)
3548 00008277 B00A                <1>         mov     al, 0Ah ; LF
3549 00008279 E804A0FFFF          <1>         call    _write_tty
3550                              <1> rw_char_retn:
3551 0000827E C3                  <1>         retn
3552                              <1>
3553                              <1> show_date:
3554                              <1>         ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
3555                              <1>         ; 2004-2005
3556                              <1>
3557                              <1>         ;mov   ah, 04h
3558                              <1>         ;call  int1Ah
3559 0000827F E86BE7FFFF          <1>         call    RTC_40 ; GET RTC DATE
3560                              <1>
3561 00008284 88D0                <1>         mov     al, dl
3562 00008286 E84B8CFFFF          <1>         call    bcd_to_ascii
3563 0000828B 66A3[C2400100]      <1>         mov     [Day], ax
3564                              <1>
3565 00008291 88F0                <1>         mov     al, dh
```

```
3566 00008293 E83E8CFFFF          <1>       call   bcd_to_ascii
3567 00008298 66A3[C5400100]      <1>       mov    [Month], ax
3568                              <1>
3569 0000829E 88E8                <1>       mov    al, ch
3570 000082A0 E8318CFFFF          <1>       call   bcd_to_ascii
3571 000082A5 66A3[C8400100]      <1>       mov    [Century], ax
3572                              <1>
3573 000082AB 88C8                <1>       mov    al, cl
3574 000082AD E8248CFFFF          <1>       call   bcd_to_ascii
3575 000082B2 66A3[CA400100]      <1>       mov    word [Year], ax
3576                              <1>
3577 000082B8 BE[B2400100]        <1>       mov    esi, Msg_Show_Date
3578 000082BD E863F1FFFF          <1>       call   print_msg
3579                              <1>
3580 000082C2 C3                  <1>       retn
3581                              <1>
3582                              <1> set_date:
3583                              <1>       ; 13/05/2016
3584                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
3585                              <1>         ; 2004-2005
3586                              <1>
3587 000082C3 BE[96400100]        <1>       mov    esi, Msg_Enter_Date
3588 000082C8 E858F1FFFF          <1>       call   print_msg
3589                              <1>
3590                              <1> loc_enter_day_1:
3591 000082CD 30E4                <1>       xor     ah, ah
3592 000082CF E80F8CFFFF          <1>       call   int16h
3593                              <1>       ; AL = ASCII Code of the Character
3594 000082D4 3C0D                <1>       cmp    al, 13
3595 000082D6 0F84B7010000        <1>       je     loc_set_date_retn
3596 000082DC 3C1B                <1>       cmp    al, 27
3597 000082DE 0F84AF010000        <1>       je     loc_set_date_retn
3598 000082E4 A2[C2400100]        <1>       mov    [Day], al
3599 000082E9 3C30                <1>       cmp    al, '0'
3600 000082EB 0F82AD010000        <1>       jb     loc_set_date_stc_0
3601 000082F1 3C33                <1>       cmp    al, '3'
3602 000082F3 0F87A5010000        <1>       ja     loc_set_date_stc_0
3603                              <1>       ; 13/05/2016
3604                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3605                              <1>                    ; video page 0 (bh)
3606 000082F9 B307                <1>       mov    bl, 7
3607 000082FB E8829FFFFF          <1>       call   _write_tty
3608                              <1> loc_enter_day_2:
3609 00008300 30E4                <1>       xor     ah, ah
3610 00008302 E8DC8BFFFF          <1>       call   int16h
3611                              <1>       ; AL = ASCII Code of the Character
3612 00008307 3C1B                <1>       cmp    al, 27
3613 00008309 0F8484010000        <1>         je      loc_set_date_retn
3614 0000830F A2[C3400100]        <1>       mov    [Day+1], al
3615 00008314 3C30                <1>       cmp    al, '0'
3616 00008316 0F828C010000        <1>         jb      loc_set_date_stc_1
3617 0000831C 3C39                <1>       cmp    al, '9'
3618 0000831E 0F8784010000        <1>         ja      loc_set_date_stc_1
3619 00008324 803D[C2400100]33    <1>       cmp    byte [Day], '3'
3620 0000832B 7208                <1>       jb     short pass_set_day_31
3621 0000832D 3C31                <1>       cmp    al, '1'
3622 0000832F 0F8773010000        <1>         ja      loc_set_date_stc_1
3623                              <1> pass_set_day_31:
3624                              <1>       ; 13/05/2016
3625                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3626                              <1>                    ; video page 0 (bh)
3627 00008335 B307                <1>       mov    bl, 7
3628 00008337 E8469FFFFF          <1>       call   _write_tty
3629                              <1> loc_enter_separator_1:
3630 0000833C 28E4                <1>       sub     ah, ah ; 0
3631 0000833E E8A08BFFFF          <1>       call   int16h
3632                              <1>       ; AL = ASCII Code of the Character
3633 00008343 3C1B                <1>       cmp    al, 27
3634 00008345 0F8448010000        <1>         je      loc_set_date_retn
3635 0000834B 3C2D                <1>       cmp    al, '-'
3636 0000834D 7408                <1>       je     short pass_set_date_separator_1
3637 0000834F 3C2F                <1>       cmp    al, '/'
3638 00008351 0F856C010000        <1>         jne     loc_set_date_stc_2
3639                              <1> pass_set_date_separator_1:
3640                              <1>       ; 13/05/2016
3641                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3642                              <1>                    ; video page 0 (bh)
3643 00008357 B307                <1>       mov    bl, 7
3644 00008359 E8249FFFFF          <1>       call   _write_tty
3645                              <1> loc_enter_month_1:
3646 0000835E 30E4                <1>       xor     ah, ah ; 0
3647 00008360 E87E8BFFFF          <1>       call   int16h
3648                              <1>       ; AL = ASCII Code of the Character
3649 00008365 3C1B                <1>       cmp    al, 27
3650 00008367 0F8426010000        <1>         je      loc_set_date_retn
3651 0000836D A2[C5400100]        <1>       mov    [Month], al
3652 00008372 3C30                <1>       cmp    al, '0'
3653 00008374 0F8264010000        <1>         jb      loc_set_date_stc_3
3654 0000837A 3C31                <1>       cmp    al, '1'
3655 0000837C 0F875C010000        <1>         ja      loc_set_date_stc_3
3656                              <1>       ; 13/05/2016
3657                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3658                              <1>                    ; video page 0 (bh)
3659 00008382 B307                <1>       mov    bl, 7
3660 00008384 E8F99EFFFF          <1>       call   _write_tty
3661                              <1> loc_enter_month_2:
3662 00008389 30E4                <1>       xor     ah, ah
3663 0000838B E8538BFFFF          <1>       call   int16h
3664                              <1>       ; AL = ASCII Code of the Character
3665 00008390 3C1B                <1>       cmp    al, 27
3666 00008392 0F84FB000000        <1>         je      loc_set_date_retn
3667 00008398 A2[C6400100]        <1>       mov    [Month+1], al
3668 0000839D 3C30                <1>       cmp    al, '0'
3669 0000839F 0F8254010000        <1>         jb      loc_set_date_stc_4
3670 000083A5 3C39                <1>       cmp    al, '9'
```

```
3671 000083A7 0F874C010000        <1>         ja      loc_set_date_stc_4
3672 000083AD 803D[C5400100]31    <1>       cmp    byte [Month], '1'
3673 000083B4 7208                <1>       jb     short pass_set_month_12
3674 000083B6 3C32                <1>       cmp    al, '2'
3675 000083B8 0F873B010000        <1>         ja      loc_set_date_stc_4
3676                              <1> pass_set_month_12:
3677                              <1>       ; 13/05/2016
3678                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3679                              <1>                    ; video page 0 (bh)
3680 000083BE B307                <1>       mov    bl, 7
3681 000083C0 E8BD9EFFFF          <1>       call   _write_tty
3682                              <1> loc_enter_separator_2:
3683 000083C5 28E4                <1>       sub     ah, ah
3684 000083C7 E8178BFFFF          <1>       call   int16h
3685                              <1>       ; AL = ASCII Code of the Character
3686 000083CC 3C1B                <1>       cmp    al, 27
3687 000083CE 0F84BF000000        <1>         je      loc_set_date_retn
3688 000083D4 3C2D                <1>       cmp    al, '-'
3689 000083D6 7408                <1>       je     short pass_set_date_separator_2
3690 000083D8 3C2F                <1>       cmp    al, '/'
3691 000083DA 0F8534010000        <1>         jne     loc_set_date_stc_5
3692                              <1> pass_set_date_separator_2:
3693                              <1>       ; 13/05/2016
3694                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3695                              <1>                      ; video page 0 (bh)
3696 000083E0 B307                <1>       mov    bl, 7
3697 000083E2 E89B9EFFFF          <1>       call   _write_tty
3698                              <1> loc_enter_year_1:
3699 000083E7 30E4                <1>       xor     ah, ah
3700 000083E9 E8F58AFFFF          <1>       call   int16h
3701                              <1>       ; AL = ASCII Code of the Character
3702 000083EE 3C1B                <1>       cmp    al, 27
3703 000083F0 0F849D000000        <1>         je      loc_set_date_retn
3704 000083F6 A2[CA400100]        <1>       mov    [Year], al
3705 000083FB 3C30                <1>       cmp    al, '0'
3706 000083FD 0F822C010000        <1>         jb      loc_set_date_stc_6
3707 00008403 3C39                <1>       cmp    al, '9'
3708 00008405 0F8724010000        <1>         ja      loc_set_date_stc_6
3709                              <1>       ; 13/05/2016
3710                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3711                              <1>                    ; video page 0 (bh)
3712 0000840B B307                <1>       mov    bl, 7
3713 0000840D E8709EFFFF          <1>       call   _write_tty
3714                              <1> loc_enter_year_2:
3715 00008412 30E4                <1>       xor    ah, ah
3716 00008414 E8CA8AFFFF          <1>       call   int16h
3717                              <1>       ; AL = ASCII Code of the Character
3718 00008419 3C1B                <1>       cmp    al, 27
3719 0000841B 7476                <1>       je     short loc_set_date_retn
3720 0000841D A2[CB400100]        <1>       mov    byte [Year+1], al
3721 00008422 3C30                <1>       cmp    al, '0'
3722 00008424 0F8220010000        <1>         jb      loc_set_date_stc_7
3723 0000842A 3C39                <1>       cmp    al, '9'
3724 0000842C 0F8718010000        <1>         ja      loc_set_date_stc_7
3725                              <1>       ; 13/05/2016
3726                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3727                              <1>                      ; video page 0 (bh)
3728 00008432 B307                <1>       mov    bl, 7
3729 00008434 E8499EFFFF          <1>       call   _write_tty
3730                              <1> loc_set_date_get_lchar_again:
3731 00008439 28E4                <1>       sub    ah, ah ; 0
3732 0000843B E8A38AFFFF          <1>       call   int16h
3733                              <1>       ; AL = ASCII Code of the Character
3734 00008440 3C0D                <1>       cmp    al, 13 ; ENTER key
3735 00008442 7412                <1>       je     short loc_set_date_progress
3736 00008444 3C1B                <1>       cmp    al, 27 ; ESC key
3737 00008446 744B                <1>       je     short loc_set_date_retn
3738                              <1>       ;
3739 00008448 E82A010000          <1>       call   check_for_backspace
3740 0000844D 75EA                <1>       jne    short loc_set_date_get_lchar_again
3741                              <1>
3742                              <1> loc_set_date_bs_8:
3743 0000844F E811010000          <1>       call   write_backspace
3744 00008454 EBBC                <1>       jmp    short loc_enter_year_2
3745                              <1>
3746                              <1> loc_set_date_progress:
3747                              <1>       ; Get Current Date
3748                              <1>       ;mov   ah, 04h
3749                              <1>       ;call  int1Ah
3750 00008456 E894E5FFFF          <1>       call   RTC_40 ; GET RTC DATE
3751                              <1>       ; CH = century (in BCD)
3752                              <1>
3753 0000845B 66A1[CA400100]      <1>       mov    ax, [Year]
3754 00008461 662D3030            <1>       sub    ax, '00'
3755 00008465 C0E004              <1>       shl    al, 4 ; * 16
3756 00008468 88C1                <1>       mov    cl, al
3757 0000846A 00E1                <1>       add    cl, ah
3758 0000846C 66A1[C5400100]      <1>       mov    ax, [Month]
3759 00008472 662D3030            <1>       sub    ax, '00'
3760 00008476 C0E004              <1>       shl    al, 4 ; * 16
3761 00008479 88C6                <1>       mov    dh, al
3762 0000847B 00E6                <1>       add    dh, ah
3763 0000847D 66A1[C2400100]      <1>       mov    ax, [Day]
3764 00008483 662D3030            <1>       sub    ax, '00'
3765 00008487 C0E004              <1>       shl    al, 4 ; * 16
3766 0000848A 88C2                <1>       mov    dl, al
3767 0000848C 00E2                <1>       add    dl, ah
3768                              <1>
3769                              <1>       ;mov   ah, 05h
3770                              <1>       ;call  int1Ah
3771 0000848E E889E5FFFF          <1>       call   RTC_50 ; SET RTC DATE
3772                              <1>
3773                              <1> loc_set_date_retn:
3774 00008493 BE[1F4C0100]        <1>       mov    esi, nextline
3775 00008498 E888EFFFFF          <1>       call   print_msg
```

```
0000849D C3                <1>       retn
                           <1>
                           <1> loc_set_date_stc_0:
                           <1>       ;xor   bh, bh ; video page 0
0000849E E8C99EFFFF        <1>       call   beeper ; BEEP !
000084A3 E925FEFFFF        <1>         jmp     loc_enter_day_1
                           <1> loc_set_date_stc_1:
000084A8 E8CA000000        <1>       call   check_for_backspace
000084AD 740A              <1>       je     short loc_set_date_bs_1
                           <1>       ;xor   bh, bh ; video page 0
000084AF E8B89EFFFF        <1>       call   beeper ; BEEP !
000084B4 E947FEFFFF        <1>         jmp     loc_enter_day_2
                           <1> loc_set_date_bs_1:
000084B9 E8A7000000        <1>       call   write_backspace
000084BE E90AFEFFFF        <1>         jmp     loc_enter_day_1
                           <1> loc_set_date_stc_2:
000084C3 E8AF000000        <1>       call   check_for_backspace
000084C8 740A              <1>       je     short loc_set_date_bs_2
                           <1>       ;xor   bh, bh ; video page 0
000084CA E89D9EFFFF        <1>       call   beeper ; BEEP !
000084CF E968FEFFFF        <1>         jmp     loc_enter_separator_1
                           <1> loc_set_date_bs_2:
000084D4 E88C000000        <1>       call   write_backspace
000084D9 E922FEFFFF        <1>         jmp     loc_enter_day_2
                           <1> loc_set_date_stc_3:
000084DE E894000000        <1>       call   check_for_backspace
000084E3 740A              <1>       je short loc_set_date_bs_3
                           <1>       ;xor   bh, bh ; video page 0
000084E5 E8829EFFFF        <1>       call   beeper ; BEEP !
000084EA E96FFEFFFF        <1>         jmp     loc_enter_month_1
                           <1> loc_set_date_bs_3:
000084EF E871000000        <1>       call   write_backspace
000084F4 E943FEFFFF        <1>         jmp     loc_enter_separator_1
                           <1> loc_set_date_stc_4:
000084F9 E879000000        <1>       call   check_for_backspace
000084FE 740A              <1>       je     short loc_set_date_bs_4
                           <1>       ;xor   bh, bh ; video page 0
00008500 E8679EFFFF        <1>       call   beeper ; BEEP !
00008505 E97FFEFFFF        <1>         jmp     loc_enter_month_2
                           <1> loc_set_date_bs_4:
0000850A E856000000        <1>       call   write_backspace
0000850F E94AFEFFFF        <1>         jmp     loc_enter_month_1
                           <1> loc_set_date_stc_5:
00008514 E85E000000        <1>       call   check_for_backspace
00008519 740A              <1>       je     short loc_set_date_bs_5
                           <1>       ;xor   bh, bh ; video page 0
0000851B E84C9EFFFF        <1>       call   beeper ; BEEP !
00008520 E9A0FEFFFF        <1>         jmp     loc_enter_separator_2
                           <1> loc_set_date_bs_5:
00008525 E83B000000        <1>       call   write_backspace
0000852A E95AFEFFFF        <1>         jmp     loc_enter_month_2
                           <1> loc_set_date_stc_6:
0000852F E843000000        <1>       call   check_for_backspace
00008534 740A              <1>         je      short  loc_set_date_bs_6
                           <1>       ;xor   bh, bh ; video page 0
00008536 E8319EFFFF        <1>       call   beeper ; BEEP !
0000853B E9A7FEFFFF        <1>         jmp     loc_enter_year_1
                           <1> loc_set_date_bs_6:
00008540 E820000000        <1>       call   write_backspace
00008545 E97BFEFFFF        <1>         jmp     loc_enter_separator_2
                           <1> loc_set_date_stc_7:
0000854A E828000000        <1>       call   check_for_backspace
0000854F 740A              <1>       je     short loc_set_date_bs_7
                           <1>       ;xor   bh, bh ; video page 0
00008551 E8169EFFFF        <1>       call   beeper ; BEEP !
00008556 E9B7FEFFFF        <1>         jmp     loc_enter_year_2
                           <1> loc_set_date_bs_7:
0000855B E805000000        <1>       call   write_backspace
00008560 E982FEFFFF        <1>         jmp     loc_enter_year_1
                           <1>
                           <1> write_backspace:
                           <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
00008565 B008              <1>       mov    al, 08h ; BACKSPACE
                           <1>       ; 13/05/2016
00008567 66BB0700          <1>       mov    bx, 7 ; bl = attribute/color
                           <1>                    ; bh = video page = 0
0000856B E8129DFFFF        <1>       call   _write_tty
00008570 B020              <1>       mov    al, 20h ; BLANK/SPACE char
                           <1>       ;mov   bx, 7 ; attribute/color
                           <1>       ;call  _write_c_current
                           <1>       ;retn
00008572 E98A9CFFFF        <1>       jmp    _write_c_current
                           <1>
                           <1> check_for_backspace:
                           <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
00008577 663D080E          <1>       cmp    ax, 0E08h
0000857B 7410              <1>       je     short cfbs_retn
0000857D 663DE04B          <1>       cmp    ax, 4BE0h
00008581 740A              <1>       je     short cfbs_retn
00008583 663D004B          <1>       cmp    ax, 4B00h
00008587 7404              <1>       je     short cfbs_retn
00008589 663DE053          <1>       cmp    ax, 53E0h
                           <1> cfbs_retn:
0000858D 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
0000858E E8EBE3FFFF        <1>       call   RTC_20 ; GET RTC TIME
                           <1>
00008593 88E8              <1>       mov    al, ch
00008595 E83C89FFFF        <1>       call   bcd_to_ascii
```

```
3881 0000859A 66A3[F0400100]      <1>       mov    [Hour], ax
3882                              <1>
3883 000085A0 88C8                <1>       mov    al, cl
3884 000085A2 E82F89FFFF          <1>       call   bcd_to_ascii
3885 000085A7 66A3[F3400100]      <1>       mov    [Minute], ax
3886                              <1>
3887 000085AD 88F0                <1>       mov    al, dh
3888 000085AF E82289FFFF          <1>       call   bcd_to_ascii
3889 000085B4 66A3[F6400100]      <1>       mov    [Second], ax
3890                              <1>
3891 000085BA BE[E0400100]        <1>       mov    esi, Msg_Show_Time
3892 000085BF E861EEFFFF          <1>       call   print_msg
3893 000085C4 C3                  <1>       retn
3894                              <1>
3895                              <1> set_time:
3896                              <1>       ; 13/05/2016
3897                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
3898                              <1>         ; 2004-2005
3899                              <1>
3900 000085C5 BE[CF400100]        <1>       mov    esi, Msg_Enter_Time
3901 000085CA E856EEFFFF          <1>       call   print_msg
3902                              <1>
3903                              <1> loc_enter_hour_1:
3904 000085CF 30E4                <1>       xor     ah, ah
3905 000085D1 E80D89FFFF          <1>       call   int16h
3906                              <1>       ; AL = ASCII Code of the Character
3907 000085D6 3C0D                <1>       cmp    al, 13 ; ENTER key
3908 000085D8 0F84AE010000        <1>         je      loc_set_time_retn
3909 000085DE 3C1B                <1>       cmp    al, 27 ; ESC key
3910 000085E0 0F84A6010000        <1>         je      loc_set_time_retn
3911 000085E6 A2[F0400100]        <1>       mov    [Hour], al
3912 000085EB 3C30                <1>       cmp    al, '0'
3913 000085ED 0F82A4010000        <1>         jb      loc_set_time_stc_0
3914 000085F3 3C32                <1>       cmp    al, '2'
3915 000085F5 0F879C010000        <1>         ja      loc_set_time_stc_0
3916                              <1>       ; 13/05/2016
3917                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3918                              <1>                    ; video page 0 (bh)
3919 000085FB B307                <1>       mov    bl, 7
3920 000085FD E8809CFFFF          <1>       call   _write_tty
3921                              <1> loc_enter_hour_2:
3922 00008602 30E4                <1>       xor     ah, ah
3923 00008604 E8DA88FFFF          <1>       call   int16h
3924                              <1>       ; AL = ASCII Code of the Character
3925 00008609 3C1B                <1>       cmp    al, 27
3926 0000860B 0F847B010000        <1>         je      loc_set_time_retn
3927 00008611 A2[F1400100]        <1>       mov    [Hour+1], al
3928 00008616 3C30                <1>       cmp    al, '0'
3929 00008618 0F8283010000        <1>         jb      loc_set_time_stc_1
3930 0000861E 3C39                <1>       cmp    al, '9'
3931 00008620 0F877B010000        <1>       ja     loc_set_time_stc_1
3932 00008626 803D[F0400100]32    <1>         cmp     byte [Hour], '2'
3933 0000862D 7208                <1>       jb     short pass_set_time_24
3934 0000862F 3C34                <1>       cmp    al, '4'
3935 00008631 0F876A010000        <1>         ja      loc_set_time_stc_1
3936                              <1> pass_set_time_24:
3937                              <1>       ; 13/05/2016
3938                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3939                              <1>                    ; video page 0 (bh)
3940 00008637 B307                <1>       mov    bl, 7
3941 00008639 E8449CFFFF          <1>       call   _write_tty
3942                              <1> loc_enter_time_separator_1:
3943 0000863E 28E4                <1>       sub     ah, ah ; 0
3944 00008640 E89E88FFFF          <1>       call   int16h
3945                              <1>       ; AL = ASCII Code of the Character
3946 00008645 3C1B                <1>       cmp    al, 27
3947 00008647 0F843F010000        <1>         je      loc_set_time_retn
3948 0000864D 3C3A                <1>       cmp    al, ':'
3949 0000864F 0F8567010000        <1>         jne     loc_set_time_stc_2
3950                              <1>       ; 13/05/2016
3951                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3952                              <1>                    ; video page 0 (bh)
3953 00008655 B307                <1>       mov    bl, 7
3954 00008657 E8269CFFFF          <1>       call   _write_tty
3955                              <1> loc_enter_minute_1:
3956 0000865C 30E4                <1>       xor     ah, ah
3957 0000865E E88088FFFF          <1>       call   int16h
3958                              <1>       ; AL = ASCII Code of the Character
3959 00008663 3C1B                <1>       cmp    al, 27
3960 00008665 0F8421010000        <1>         je      loc_set_time_retn
3961 0000866B A2[F3400100]        <1>       mov    [Minute], al
3962 00008670 3C30                <1>       cmp    al, '0'
3963 00008672 0F825F010000        <1>         jb      loc_set_time_stc_3
3964 00008678 3C35                <1>       cmp    al, '5'
3965 0000867A 0F8757010000        <1>         ja      loc_set_time_stc_3
3966                              <1>       ; 13/05/2016
3967                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3968                              <1>                    ; video page 0 (bh)
3969 00008680 B307                <1>       mov    bl, 7
3970 00008682 E8FB9BFFFF          <1>       call   _write_tty
3971                              <1> loc_enter_minute_2:
3972 00008687 30E4                <1>       xor     ah, ah
3973 00008689 E85588FFFF          <1>       call   int16h
3974                              <1>       ; AL = ASCII Code of the Character
3975 0000868E 3C1B                <1>       cmp    al, 27
3976 00008690 0F84F6000000        <1>         je      loc_set_time_retn
3977 00008696 A2[F4400100]        <1>       mov    [Minute+1], al
3978 0000869B 3C30                <1>       cmp    al, '0'
3979 0000869D 0F824F010000        <1>         jb      loc_set_time_stc_4
3980 000086A3 3C39                <1>       cmp    al, '9'
3981 000086A5 0F8747010000        <1>         ja      loc_set_time_stc_4
3982                              <1>       ; 13/05/2016
3983                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3984                              <1>                    ; video page 0 (bh)
3985 000086AB B307                <1>       mov    bl, 7
```

```
3986 000086AD E8D09BFFFF          <1>       call   _write_tty
3987                              <1> loc_enter_time_separator_2:
3988 000086B2 66C705[F6400100]30- <1>       mov    word [Second], 3030h
3988 000086BA 30                  <1>
3989 000086BB 28E4                <1>       sub     ah, ah
3990 000086BD E82188FFFF          <1>       call   int16h
3991                              <1>       ; AL = ASCII Code of the Character
3992 000086C2 3C0D                <1>       cmp    al, 13
3993 000086C4 0F8485000000        <1>         je      loc_set_time_progress
3994 000086CA 3C1B                <1>       cmp    al, 27
3995 000086CC 0F84BA000000        <1>         je      loc_set_time_retn
3996 000086D2 3C3A                <1>       cmp    al, ':'
3997 000086D4 0F8533010000        <1>         jne     loc_set_time_stc_5
3998                              <1>       ; 13/05/2016
3999                              <1>       ;mov   bx, 7 ; attribute/color (bl)
4000                              <1>                    ; video page 0 (bh)
4001 000086DA B307                <1>       mov    bl, 7
4002 000086DC E8A19BFFFF          <1>       call   _write_tty
4003                              <1> loc_enter_second_1:
4004 000086E1 30E4                <1>       xor     ah, ah
4005 000086E3 E8FB87FFFF          <1>       call   int16h
4006                              <1>       ; AL = ASCII Code of the Character
4007 000086E8 3C0D                <1>       cmp    al, 13
4008 000086EA 7463                <1>       je     short loc_set_time_progress
4009 000086EC 3C1B                <1>       cmp    al, 27
4010 000086EE 0F8498000000        <1>         je      loc_set_time_retn
4011 000086F4 A2[F6400100]        <1>       mov    [Second], al
4012 000086F9 3C30                <1>       cmp    al, '0'
4013 000086FB 0F8227010000        <1>         jb      loc_set_time_stc_6
4014 00008701 3C35                <1>       cmp    al, '5'
4015 00008703 0F871F010000        <1>         ja      loc_set_time_stc_6
4016                              <1>       ; 13/05/2016
4017                              <1>       ;mov   bx, 7 ; attribute/color (bl)
4018                              <1>                    ; video page 0 (bh)
4019 00008709 B307                <1>       mov    bl, 7
4020 0000870B E8729BFFFF          <1>       call   _write_tty
4021                              <1> loc_enter_second_2:
4022 00008710 30E4                <1>       xor     ah, ah
4023 00008712 E8CC87FFFF          <1>       call   int16h
4024                              <1>       ; AL = ASCII Code of the Character
4025 00008717 3C1B                <1>       cmp    al, 27
4026 00008719 7471                <1>       je     short loc_set_time_retn
4027 0000871B 3C30                <1>       cmp    al, '0'
4028 0000871D 0F8229010000        <1>         jb      loc_set_time_stc_7
4029 00008723 3C39                <1>       cmp    al, '9'
4030 00008725 0F8721010000        <1>         ja      loc_set_time_stc_7
4031                              <1>       ; 13/05/2016
4032                              <1>       ;mov   bx, 7 ; attribute/color (bl)
4033                              <1>                    ; video page 0 (bh)
4034 0000872B B307                <1>       mov    bl, 7
4035 0000872D E8509BFFFF          <1>       call   _write_tty
4036                              <1> loc_set_time_get_lchar_again:
4037 00008732 28E4                <1>       sub    ah, ah ; 0
4038 00008734 E8AA87FFFF          <1>       call   int16h
4039                              <1>       ; AL = ASCII Code of the Character
4040 00008739 3C0D                <1>       cmp    al, 13
4041 0000873B 7412                <1>       je     short loc_set_time_progress
4042 0000873D 3C1B                <1>       cmp    al, 27
4043 0000873F 744B                <1>       je     short loc_set_time_retn
4044                              <1>       ;
4045 00008741 E831FEFFFF          <1>       call   check_for_backspace
4046 00008746 75EA                <1>       jne    short loc_set_time_get_lchar_again
4047                              <1>
4048                              <1> loc_set_time_bs_8:
4049 00008748 E818FEFFFF          <1>       call   write_backspace
4050 0000874D EBC1                <1>       jmp    short loc_enter_second_2
4051                              <1>
4052                              <1> loc_set_time_progress:
4053                              <1>       ; Get Current Time
4054                              <1>       ;mov   ah, 02h
4055                              <1>       ;call  int1Ah
4056 0000874F E82AE2FFFF          <1>       call   RTC_20 ; GET RTC TIME
4057                              <1>       ;DL = Daylight Savings Enable option (0-1)
4058                              <1>
4059 00008754 66A1[F0400100]      <1>       mov    ax, [Hour]
4060 0000875A 662D3030            <1>       sub    ax, '00'
4061 0000875E C0E004              <1>       shl    al, 4 ; * 16
4062 00008761 88C5                <1>       mov    ch, al
4063 00008763 00E5                <1>       add    ch, ah
4064 00008765 66A1[F3400100]      <1>       mov    ax, [Minute]
4065 0000876B 662D3030            <1>       sub    ax, '00'
4066 0000876F C0E004              <1>       shl    al, 4 ; * 16
4067 00008772 88C1                <1>       mov    cl, al
4068 00008774 00E1                <1>       add    cl, ah
4069 00008776 66A1[F6400100]      <1>       mov    ax, [Second]
4070 0000877C 662D3030            <1>       sub    ax, '00'
4071 00008780 C0E004              <1>       shl    al, 4 ; * 16
4072 00008783 88C6                <1>       mov    dh, al
4073 00008785 00E6                <1>       add    dh, ah
4074                              <1>
4075                              <1>       ;mov   ah, 03h
4076                              <1>       ;call  int1Ah
4077 00008787 E821E2FFFF          <1>       call   RTC_30 ; SET RTC TIME
4078                              <1>
4079                              <1> loc_set_time_retn:
4080 0000878C BE[1F4C0100]        <1>       mov    esi, nextline
4081 00008791 E88FECFFFF          <1>       call   print_msg
4082 00008796 C3                  <1>       retn
4083                              <1>
4084                              <1> loc_set_time_stc_0:
4085                              <1>       ;xor   bh, bh ; video page 0
4086 00008797 E8D09BFFFF          <1>       call   beeper ; BEEP !
4087 0000879C E92EFEFFFF          <1>         jmp     loc_enter_hour_1
4088                              <1> loc_set_time_stc_1:
4089 000087A1 E8D1FDFFFF          <1>       call   check_for_backspace
```

```
4090 000087A6 740A              <1>       je     short loc_set_time_bs_1
4091                            <1>       ;xor   bh, bh ; video page 0
4092 000087A8 E8BF9BFFFF        <1>       call   beeper ; BEEP !
4093 000087AD E950FEFFFF        <1>         jmp     loc_enter_hour_2
4094                            <1> loc_set_time_bs_1:
4095 000087B2 E8AEFDFFFF        <1>       call   write_backspace
4096 000087B7 E913FEFFFF        <1>         jmp     loc_enter_hour_1
4097                            <1> loc_set_time_stc_2:
4098 000087BC E8B6FDFFFF        <1>       call   check_for_backspace
4099 000087C1 740A              <1>       je     short loc_set_time_bs_2
4100                            <1>       ;xor   bh, bh ; video page 0
4101 000087C3 E8A49BFFFF        <1>       call   beeper ; BEEP !
4102 000087C8 E971FEFFFF        <1>         jmp     loc_enter_time_separator_1
4103                            <1> loc_set_time_bs_2:
4104 000087CD E893FDFFFF        <1>       call   write_backspace
4105 000087D2 E92BFEFFFF        <1>         jmp     loc_enter_hour_2
4106                            <1> loc_set_time_stc_3:
4107 000087D7 E89BFDFFFF        <1>       call   check_for_backspace
4108 000087DC 740A              <1>       je     short loc_set_time_bs_3
4109                            <1>       ;xor   bh, bh ; video page 0
4110 000087DE E8899BFFFF        <1>       call   beeper ; BEEP !6
4111 000087E3 E974FEFFFF        <1>         jmp     loc_enter_minute_1
4112                            <1> loc_set_time_bs_3:
4113 000087E8 E878FDFFFF        <1>       call   write_backspace
4114 000087ED E94CFEFFFF        <1>         jmp     loc_enter_time_separator_1
4115                            <1> loc_set_time_stc_4:
4116 000087F2 E880FDFFFF        <1>       call   check_for_backspace
4117 000087F7 740A              <1>       je     short loc_set_time_bs_4
4118                            <1>       ;xor   bh, bh ; video page 0
4119 000087F9 E86E9BFFFF        <1>       call   beeper ; BEEP !
4120 000087FE E984FEFFFF        <1>         jmp     loc_enter_minute_2
4121                            <1> loc_set_time_bs_4:
4122 00008803 E85DFDFFFF        <1>       call   write_backspace
4123 00008808 E94FFEFFFF        <1>         jmp     loc_enter_minute_1
4124                            <1> loc_set_time_stc_5:
4125 0000880D E865FDFFFF        <1>       call   check_for_backspace
4126 00008812 740A              <1>       je     short loc_set_time_bs_5
4127                            <1>       ;xor   bh, bh ; video page 0
4128 00008814 E8539BFFFF        <1>       call   beeper ; BEEP !
4129 00008819 E994FEFFFF        <1>         jmp     loc_enter_time_separator_2
4130                            <1> loc_set_time_bs_5:
4131 0000881E E842FDFFFF        <1>       call   write_backspace
4132 00008823 E95FFEFFFF        <1>         jmp     loc_enter_minute_2
4133                            <1> loc_set_time_stc_6:
4134 00008828 E84AFDFFFF        <1>       call   check_for_backspace
4135 0000882D 7413              <1>       je     short loc_set_time_bs_6
4136                            <1>       ;xor   bh, bh ; video page 0
4137 0000882F E8389BFFFF        <1>       call   beeper ; BEEP !
4138 00008834 66C705[F6400100]30- <1>     mov    word [Second], 3030h
4138 0000883C 30                <1>
4139 0000883D E99FFEFFFF        <1>         jmp     loc_enter_second_1
4140                            <1> loc_set_time_bs_6:
4141 00008842 E81EFDFFFF        <1>       call   write_backspace
4142 00008847 E966FEFFFF        <1>         jmp     loc_enter_time_separator_2
4143                            <1> loc_set_time_stc_7:
4144 0000884C E826FDFFFF        <1>       call   check_for_backspace
4145 00008851 740A              <1>       je     short loc_set_time_bs_7
4146                            <1>       ;xor   bh, bh ; video page 0
4147 00008853 E8149BFFFF        <1>       call   beeper ; BEEP !
4148 00008858 E9B3FEFFFF        <1>         jmp     loc_enter_second_2
4149                            <1> loc_set_time_bs_7:
4150 0000885D E803FDFFFF        <1>       call   write_backspace
4151 00008862 E97AFEFFFF        <1>         jmp     loc_enter_second_1
4152                            <1>
4153                            <1> print_volume_info:
4154                            <1>       ; 01/03/2016
4155                            <1>       ; 08/02/2016
4156                            <1>       ; 06/02/2016
4157                            <1>       ; 04/02/2016
4158                            <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
4159                            <1>       ; 25/10/2009
4160                            <1>       ;
4161                            <1>       ; "Volume Serial No: "
4162                            <1>       ;
4163                            <1>       ; INPUT  : AL = DOS Drive Number
4164                            <1>       ; OUTPUT : AH = FS Type
4165                            <1>       ;          AL = DOS Drive Name
4166                            <1>       ; CF = 0 -> OK
4167                            <1>       ; CF = 1 -> Drive not ready
4168                            <1>
4169 00008867 88C4              <1>       mov    ah, al
4170 00008869 28C0              <1>       sub    al, al
4171 0000886B 0FB7F0            <1>       movzx  esi, ax
4172 0000886E 81C600010900      <1>       add    esi, Logical_DOSDisks
4173 00008874 8A06              <1>       mov    al, [esi]
4174 00008876 3C41              <1>       cmp    al, 'A'
4175 00008878 7304              <1>       jnb    short loc_pvi_set_vol_name
4176 0000887A 8A6604            <1>       mov    ah, [esi+LD_FSType]
4177 0000887D C3                <1>       retn
4178                            <1>
4179                            <1> loc_pvi_set_vol_name:
4180 0000887E A2[2A410100]      <1>       mov    [Vol_Drv_Name], al
4181 00008883 56                <1>       push   esi
4182 00008884 E858010000        <1>       call   move_volume_name_and_serial_no ;;;
4183 00008889 7302              <1>       jnc    short loc_pvi_mvn_ok
4184 0000888B 5E                <1>       pop    esi
4185 0000888C C3                <1>       retn
4186                            <1>
4187                            <1> loc_pvi_mvn_ok:
4188 0000888D 8B3424            <1>       mov    esi, [esp]
4189 00008890 807E04A1          <1>       cmp    byte [esi+LD_FSType], 0A1h
4190 00008894 7509              <1>       jne    short loc_pvi_fat_vol_size
4191 00008896 8B4670            <1>       mov    eax, [esi+LD_FS_VolumeSize]
4192 00008899 0FB75E11          <1>       movzx  ebx, word [esi+LD_FS_BytesPerSec]
4193 0000889D EB07              <1>       jmp    short loc_vol_size_mul32
```

```
                                <1> loc_pvi_fat_vol_size:
0000889F 8B4670                 <1>     mov     eax, [esi+LD_TotalSectors]
000088A2 0FB75E11               <1>     movzx   ebx, word [esi+LD_BPB+BPB_BytsPerSec]
                                <1> loc_vol_size_mul32:
000088A6 F7E3                   <1>     mul     ebx
000088A8 09D2                   <1>     or      edx, edx
000088AA 7507                   <1>     jnz     short loc_vol_size_in_kbytes
                                <1> loc_vol_size_in_bytes:
000088AC B9[08410100]           <1>     mov     ecx, VolSize_Bytes
000088B1 EB0D                   <1>     jmp     short loc_write_vol_size_str
                                <1> loc_vol_size_in_kbytes:
000088B3 66BB0004               <1>     mov     bx, 1024
000088B7 F7F3                   <1>     div     ebx
000088B9 B9[FB400100]           <1>     mov     ecx, VolSize_KiloBytes
000088BE 31D2                   <1>     xor     edx, edx ; 0
                                <1> loc_write_vol_size_str:
000088C0 890D[EB900100]         <1>     mov     [VolSize_Unit1], ecx
                                <1>     ;
000088C6 BF[01910100]           <1>     mov     edi, Vol_Tot_Sec_Str_End
                                <1>       ;mov  byte [edi], 0
000088CB B90A000000             <1>     mov     ecx, 10
                                <1> loc_write_vol_size_chr:
000088D0 F7F1                   <1>     div     ecx
000088D2 80C230                 <1>     add     dl, '0'
000088D5 4F                     <1>     dec     edi
000088D6 8817                   <1>     mov     [edi], dl
000088D8 85C0                   <1>     test    eax, eax
000088DA 7404                   <1>     jz      short loc_write_vol_size_str_ok
000088DC 28D2                   <1>     sub     dl, dl ; 0
000088DE EBF0                   <1>     jmp     short loc_write_vol_size_chr
                                <1>
                                <1> loc_write_vol_size_str_ok:
000088E0 893D[F3900100]         <1>     mov     [Vol_Tot_Sec_Str_Start], edi
                                <1>     ;
000088E6 BF[13410100]           <1>     mov     edi, Vol_FS_Name
000088EB 8A4E03                 <1>     mov     cl, [esi+LD_FATType]
000088EE 20C9                   <1>     and     cl, cl ; 0 ?
000088F0 7515                   <1>     jnz     short loc_write_vol_FAT_str_1
000088F2 66C7075452             <1>     mov     word [edi], 'TR'
000088F7 C7470420465331         <1>     mov     dword [edi+4], ' FS1'
                                <1>     ;movzx  ebx, word [esi+LD_FS_BytesPerSec]
000088FE 668B5E11               <1>     mov     bx, [esi+LD_FS_BytesPerSec]
00008902 8B4674                 <1>     mov     eax, [esi+LD_FS_FreeSectors]
00008905 EB36                   <1>     jmp     short loc_vol_freespace_mul32
                                <1>
                                <1> loc_write_vol_FAT_str_1:
00008907 66B83332               <1>     mov     ax, '32' ; FAT32
0000890B 80F902                 <1>     cmp     cl, 2 ; [esi+LD_FATType]
0000890E 7708                   <1>     ja      short loc_write_vol_FAT_str_2
00008910 66B83132               <1>     mov     ax, '12' ; FAT12
00008914 7202                   <1>     jb      short loc_write_vol_FAT_str_2
00008916 B436                   <1>     mov     ah, '6'  ; FAT16
                                <1> loc_write_vol_FAT_str_2:
00008918 C70746415420           <1>     mov     dword [edi], 'FAT '
0000891E 66894704               <1>     mov     word [edi+4], ax
                                <1>     ;
                                <1>     ;movzx  ebx, word [esi+LD_BPB+BPB_BytsPerSec]
00008922 668B5E11               <1>     mov     bx, [esi+LD_BPB+BPB_BytsPerSec]
00008926 8B4674                 <1>     mov     eax, [esi+LD_FreeSectors]
                                <1>
                                <1> loc_vol_freespace_recalc0:
                                <1>     ; 01/03/2016
00008929 83F8FF                 <1>     cmp     eax, 0FFFFFFFFh
0000892C 720F                   <1>     jb      short loc_vol_freespace_mul32
                                <1>     ;inc    eax ; 0
0000892E 20C9                   <1>     and     cl, cl ; byte [esi+LD_FATType]
00008930 740B                   <1>     jz      short loc_vol_freespace_mul32
00008932 53                     <1>     push    ebx
00008933 66BB00FF               <1>     mov     bx, 0FF00h ; recalculate free sectors
00008937 E876490000             <1>     call    calculate_fat_freespace
0000893C 5B                     <1>     pop     ebx
                                <1>
                                <1> loc_vol_freespace_mul32:
0000893D F7E3                   <1>     mul     ebx
0000893F 09D2                   <1>     or      edx, edx
00008941 7507                   <1>     jnz     short loc_vol_fspace_in_kbytes
                                <1> loc_vol_fspace_in_bytes:
00008943 B9[08410100]           <1>     mov     ecx, VolSize_Bytes
00008948 EB0D                   <1>     jmp     short loc_write_vol_fspace_str
                                <1> loc_vol_fspace_in_kbytes:
0000894A 66BB0004               <1>     mov     bx, 1024
0000894E F7F3                   <1>     div     ebx
00008950 B9[FB400100]           <1>     mov     ecx, VolSize_KiloBytes
00008955 31D2                   <1>     xor     edx, edx ; 0
                                <1> loc_write_vol_fspace_str:
00008957 890D[EF900100]         <1>     mov     [VolSize_Unit2], ecx
                                <1>     ;
0000895D BF[11910100]           <1>     mov     edi, Vol_Free_Sectors_Str_End
                                <1>       ;mov  byte [edi], 0
00008962 B90A000000             <1>     mov     ecx, 10
                                <1> loc_write_vol_fspace_chr:
00008967 F7F1                   <1>     div     ecx
00008969 80C230                 <1>     add     dl, '0'
0000896C 4F                     <1>     dec     edi
0000896D 8817                   <1>     mov     [edi], dl
0000896F 85C0                   <1>     test    eax, eax
00008971 7404                   <1>     jz      short loc_write_vol_fspace_str_ok
00008973 28D2                   <1>     sub     dl, dl ; 0
00008975 EBF0                   <1>     jmp     short loc_write_vol_fspace_chr
                                <1>
                                <1> loc_write_vol_fspace_str_ok:
00008977 893D[03910100]         <1>     mov     [Vol_Free_Sectors_Str_Start], edi
                                <1>     ;
0000897D BE[11410100]           <1>     mov     esi, Volume_in_drive
00008982 E89EEAFFFF             <1>     call    print_msg
```

```
4299 00008987 BE[51410100]        <1>       mov   esi, Vol_Name
4300 0000898C E894EAFFFF          <1>       call  print_msg
4301 00008991 BE[1F4C0100]        <1>       mov   esi, nextline
4302 00008996 E88AEAFFFF          <1>       call  print_msg
4303                              <1>       ;
4304 0000899B BE[B2410100]        <1>       mov   esi, Vol_Total_Sector_Header
4305 000089A0 E880EAFFFF          <1>       call  print_msg
4306 000089A5 8B35[F3900100]      <1>       mov   esi, [Vol_Tot_Sec_Str_Start]
4307 000089AB E875EAFFFF          <1>       call  print_msg
4308 000089B0 8B35[EB900100]      <1>       mov   esi, [VolSize_Unit1]
4309 000089B6 E86AEAFFFF          <1>       call  print_msg
4310                              <1>       ;
4311 000089BB BE[C3410100]        <1>       mov   esi, Vol_Free_Sectors_Header
4312 000089C0 E860EAFFFF          <1>       call  print_msg
4313 000089C5 8B35[03910100]      <1>       mov   esi, [Vol_Free_Sectors_Str_Start]
4314 000089CB E855EAFFFF          <1>       call  print_msg
4315 000089D0 8B35[EF900100]      <1>       mov   esi, [VolSize_Unit2]
4316 000089D6 E84AEAFFFF          <1>       call  print_msg
4317                              <1>       ;
4318 000089DB 5E                  <1>       pop   esi
4319                              <1>
4320                              <1>       ;mov  ah, [esi+LD_FSType]
4321                              <1>       ;mov  al, [esi+LD_FATType]
4322 000089DC 668B4603            <1>       mov   ax, [esi+LD_FATType]
4323                              <1>
4324 000089E0 C3                  <1>       retn
4325                              <1>
4326                              <1> move_volume_name_and_serial_no:
4327                              <1>       ; 08/02/2016  (TRDOS 386 = TRDOS v2.0)
4328                              <1>       ; this routine will be called by
4329                              <1>       ; "print_volume_info" and "print_directory"
4330                              <1>       ; INPUT ->
4331                              <1>       ;     ESI = Logical DOS drv descripton table address
4332                              <1>       ; OUTPUT ->
4333                              <1>       ;     *Volume name will be moved to text area
4334                              <1>       ;     *Volume serial number will be converted to
4335                              <1>       ;      text and will be moved to text area
4336                              <1>       ;   cf = 1 -> invalid/unknown dos drive
4337                              <1>       ;   cf = 0 -> ecx = 0
4338                              <1>       ;
4339                              <1>       ; (eax, edx, ecx, esi, edi will be changed)
4340                              <1>
4341 000089E1 BF[51410100]        <1>       mov   edi, Vol_Name
4342                              <1>
4343                              <1>       ;mov  ah, [esi+LD_FSType]
4344                              <1>       ;mov  al, [esi+LD_FATType]
4345 000089E6 668B4603            <1>       mov   ax, [esi+LD_FATType]
4346 000089EA 80FCA1              <1>       cmp   ah, 0A1h
4347 000089ED 7418                <1>       je    short mvn_2
4348 000089EF 08E4                <1>       or    ah, ah
4349 000089F1 7404                <1>       jz    short mvn_0
4350 000089F3 08C0                <1>       or    al, al
4351 000089F5 7504                <1>       jnz   short mvn_1
4352                              <1> mvn_0:
4353 000089F7 8A06                <1>       mov   al, [esi]
4354 000089F9 F9                  <1>       stc
4355 000089FA C3                  <1>       retn
4356                              <1> mvn_1:
4357 000089FB 3C02                <1>       cmp   al, 2
4358 000089FD 7717                <1>       ja    short mvn_3
4359                              <1>       ;or   al, al
4360                              <1>       ;jz   short mvn_2
4361 000089FF 8B462D              <1>       mov   eax, [esi+LD_BPB+VolumeID]
4362 00008A02 83C631              <1>       add   esi, LD_BPB+VolumeLabel
4363 00008A05 EB15                <1>       jmp   short mvn_4
4364                              <1> mvn_2:
4365 00008A07 8B4628              <1>       mov   eax, [esi+LD_FS_VolumeSerial]
4366 00008A0A 83C62C              <1>       add   esi, LD_FS_VolumeName
4367 00008A0D B910000000          <1>       mov   ecx, 16
4368 00008A12 F3A5                <1>       rep   movsd
4369 00008A14 EB10                <1>       jmp   short mvn_5
4370                              <1> mvn_3:
4371 00008A16 8B4649              <1>       mov   eax, [esi+LD_BPB+FAT32_VolID]
4372 00008A19 83C64D              <1>       add   esi, LD_BPB+FAT32_VolLab
4373                              <1> mvn_4:
4374 00008A1C B90B000000          <1>       mov   ecx, 11
4375 00008A21 F3A4                <1>       rep   movsb
4376 00008A23 C60700              <1>       mov   byte [edi], 0
4377                              <1> mvn_5:
4378                              <1>       ;mov  [Current_VolSerial], eax
4379 00008A26 E845B8FFFF          <1>       call  dwordtohex
4380 00008A2B 8915[A6410100]      <1>       mov   [Vol_Serial1], edx
4381 00008A31 A3[AB410100]        <1>       mov   [Vol_Serial2], eax
4382                              <1>       ; ecx = 0
4383 00008A36 C3                  <1>       retn
4384                              <1>
4385                              <1> get_volume_serial_number:
4386                              <1>       ; 19/01/2016 (TRDOS 386 = TRDOS v2.0)
4387                              <1>       ; 08/08/2010
4388                              <1>       ;
4389                              <1>       ; INPUT -> DL = Logical DOS Drive number
4390                              <1>       ; OUTPUT -> EAX = Volume serial number
4391                              <1>       ;          BL= FAT Type
4392                              <1>       ;          BH = Logical DOS drv Number (DL input)
4393                              <1>       ; cf = 1 -> Drive not ready
4394                              <1>
4395 00008A37 31DB                <1>       xor   ebx, ebx
4396 00008A39 88D7                <1>       mov   bh, dl
4397 00008A3B 3815[823F0100]      <1>       cmp   [Last_DOS_DiskNo], dl
4398 00008A41 7304                <1>       jnb   short loc_gvsn_start
4399                              <1> loc_gvsn_stc_retn:
4400 00008A43 31C0                <1>       xor   eax, eax
4401 00008A45 F9                  <1>       stc
4402 00008A46 C3                  <1>         retn
4403                              <1> loc_gvsn_start:
```

```
4404 00008A47 56                <1>       push   esi
4405 00008A48 BE00010900        <1>       mov    esi, Logical_DOSDisks
4406 00008A4D 01DE              <1>       add    esi, ebx
4407 00008A4F 8A5E03            <1>       mov    bl, [esi+LD_FATType]
4408 00008A52 20DB              <1>       and    bl, bl
4409 00008A54 740F              <1>       jz     short loc_gvsn_fs
4410 00008A56 80FB02            <1>       cmp    bl, 2
4411 00008A59 7705              <1>       ja     short loc_gvsn_fat32
4412                            <1> loc_gvsn_fat:
4413 00008A5B 83C62D            <1>       add    esi, LD_BPB + VolumeID
4414 00008A5E EB0E              <1>       jmp    short loc_gvsn_return
4415                            <1> loc_gvsn_fat32:
4416 00008A60 83C649            <1>       add    esi, LD_BPB + FAT32_VolID
4417 00008A63 EB09              <1>       jmp    short loc_gvsn_return
4418                            <1> loc_gvsn_fs:
4419 00008A65 807E04A1          <1>       cmp    byte [esi+LD_FSType], 0A1h
4420 00008A69 75D8              <1>       jne    short loc_gvsn_stc_retn
4421 00008A6B 83C628            <1>       add    esi, LD_FS_VolumeSerial
4422                            <1> loc_gvsn_return:
4423 00008A6E 8B06              <1>       mov    eax, [esi]
4424 00008A70 5E                <1>       pop    esi
4425 00008A71 C3                <1>       retn
4426                            <1>
4427                            <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ]
4428                            <1> ; 09/11/2011
4429                            <1> ; 29/01/2005
4430                            <1>
4431                            <1> command_interpreter:
4432                            <1>       ; 16/10/2016
4433                            <1>       ; 12/10/2016
4434                            <1>       ; 13/05/2016
4435                            <1>       ; 07/05/2016
4436                            <1>       ; 04/03/2016
4437                            <1>       ; 04/02/2016
4438                            <1>       ; 03/02/2016
4439                            <1>       ; 30/01/2016
4440                            <1>       ; 29/01/2016 (TRDOS 386 = TRDOS 2.0)
4441                            <1>       ; 15/09/2011
4442                            <1>       ; 29/01/2005
4443                            <1>
4444                            <1>       ; Input: ecx = command word length (CL)
4445                            <1>       ;      CommandBuffer = Command string offset
4446                            <1>
4447 00008A72 C605[A4910100]00  <1>       mov    byte [Program_Exit],0
4448 00008A79 80F904            <1>       cmp    cl, 4
4449 00008A7C 0F87B5020000      <1>         ja     c_6
4450 00008A82 0F8237010000      <1>         jb     c_2
4451                            <1> c_4:
4452                            <1>
4453                            <1> cmp_cmd_exit:
4454 00008A88 BF[F03F0100]      <1>       mov    edi, Cmd_Exit
4455 00008A8D E8C2030000        <1>       call   cmp_cmd
4456 00008A92 7208              <1>       jc     short cmp_cmd_date
4457                            <1>
4458 00008A94 C605[A4910100]01  <1>         mov    byte [Program_Exit], 1
4459 00008A9B C3                <1>         retn
4460                            <1>
4461                            <1> cmp_cmd_date:
4462 00008A9C B104              <1>       mov    cl, 4
4463 00008A9E BF[0C400100]      <1>       mov    edi, Cmd_Date
4464 00008AA3 E8AC030000        <1>       call   cmp_cmd
4465 00008AA8 720B              <1>         jc   short cmp_cmd_time
4466                            <1>
4467 00008AAA E8D0F7FFFF        <1>       call   show_date
4468 00008AAF E80FF8FFFF        <1>       call   set_date
4469 00008AB4 C3                <1>       retn
4470                            <1>
4471                            <1> cmp_cmd_time:
4472 00008AB5 B104              <1>       mov    cl, 4
4473 00008AB7 BF[11400100]      <1>       mov    edi, Cmd_Time
4474 00008ABC E893030000        <1>       call   cmp_cmd
4475 00008AC1 720B              <1>       jc     short cmp_cmd_show
4476                            <1>
4477 00008AC3 E8C6FAFFFF        <1>       call   show_time
4478 00008AC8 E8F8FAFFFF        <1>       call   set_time
4479 00008ACD C3                <1>       retn
4480                            <1>
4481                            <1> cmp_cmd_show:
4482 00008ACE B104              <1>       mov    cl, 4
4483 00008AD0 BF[22400100]      <1>       mov    edi, Cmd_Show
4484 00008AD5 E87A030000        <1>       call   cmp_cmd
4485 00008ADA 0F83050A0000      <1>         jnc     show_file
4486                            <1>
4487                            <1> cmp_cmd_echo:
4488 00008AE0 B104              <1>       mov    cl, 4
4489 00008AE2 BF[5E400100]      <1>       mov    edi, Cmd_Echo
4490 00008AE7 E868030000        <1>       call   cmp_cmd
4491 00008AEC 7224              <1>       jc     short cmp_cmd_copy
4492                            <1>
4493                            <1>       ; 22/11/2017
4494                            <1>       ; AL = 0
4495 00008AEE 803E20            <1>       cmp    byte [esi], 20h
4496 00008AF1 7215              <1>       jb     short cmd_echo_nextline
4497                            <1>       ; 14/04/2016
4498 00008AF3 56                <1>       push   esi
4499                            <1> cmd_echo_asciiz:
4500                            <1>       ;inc   esi
4501                            <1>       ;mov   al, [esi]
4502                            <1>       ; 22/11/2017
4503 00008AF4 AC                <1>       lodsb
4504 00008AF5 3C20              <1>       cmp    al, 20h
4505 00008AF7 73FB              <1>       jnb    short cmd_echo_asciiz
4506 00008AF9 4E                <1>       dec    esi
4507 00008AFA C60600            <1>       mov    byte [esi], 0
4508 00008AFD 5E                <1>       pop    esi
```

```
4509 00008AFE 89F7              <1>       mov    edi, esi
4510 00008B00 E820E9FFFF        <1>       call   print_msg
4511 00008B05 C60700            <1>       mov    byte [edi], 0
4512                            <1> cmd_echo_nextline:
4513 00008B08 BE[8D4C0100]      <1>       mov    esi, NextLine
4514                            <1>       ;call  print_msg
4515                            <1>       ;retn
4516 00008B0D E913E9FFFF        <1>       jmp    print_msg
4517                            <1>
4518                            <1> cmp_cmd_copy:
4519 00008B12 B104              <1>       mov    cl, 4
4520 00008B14 BF[45400100]      <1>       mov    edi, Cmd_Copy
4521 00008B19 E836030000        <1>       call   cmp_cmd
4522 00008B1E 0F83CC170000      <1>       jnc    copy_file
4523                            <1>
4524                            <1> cmp_cmd_move:
4525 00008B24 B104              <1>       mov    cl, 4
4526 00008B26 BF[4A400100]      <1>       mov    edi, Cmd_Move
4527 00008B2B E824030000        <1>       call   cmp_cmd
4528 00008B30 0F836E160000      <1>       jnc    move_file
4529                            <1>
4530                            <1> cmp_cmd_path:
4531 00008B36 B104              <1>       mov    cl, 4
4532 00008B38 BF[4F400100]      <1>       mov    edi, Cmd_Path
4533 00008B3D E812030000        <1>       call   cmp_cmd
4534 00008B42 0F83F0190000      <1>       jnc    set_get_path
4535                            <1>
4536                            <1> cmp_cmd_beep:
4537 00008B48 B104              <1>       mov    cl, 4
4538 00008B4A BF[7C400100]      <1>       mov    edi, Cmd_Beep
4539 00008B4F E800030000        <1>       call   cmp_cmd
4540 00008B54 720B              <1>       jc     short cmp_cmd_find
4541                            <1>       ; 13/05/2016
4542 00008B56 8A3D[1E890100]    <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
4543 00008B5C E90B98FFFF        <1>       jmp    beeper
4544                            <1>
4545                            <1> cmp_cmd_find:
4546 00008B61 B104              <1>       mov    cl, 4
4547 00008B63 BF[59400100]      <1>       mov    edi, Cmd_Find
4548 00008B68 E8E7020000        <1>       call   cmp_cmd
4549 00008B6D 0F82C4020000      <1>         jc     cmp_cmd_external
4550                            <1>
4551                            <1>       ;call  find_and_list_files
4552 00008B73 E9AF220000        <1>       jmp    find_and_list_files
4553                            <1>       ;retn
4554                            <1>
4555                            <1> c_1:
4556 00008B78 AD                <1>       lodsd
4557                            <1> cmp_cmd_help:
4558 00008B79 3C3F              <1>       cmp    al, '?'
4559 00008B7B 751D              <1>         jne    short cmp_cmd_remark
4560                            <1>
4561 00008B7D BE[E23F0100]      <1>       mov    esi, Command_List
4562                            <1> cmd_help_next_w:
4563 00008B82 E89EE8FFFF        <1>       call   print_msg
4564                            <1>
4565 00008B87 803E20            <1>       cmp    byte [esi], 20h ; 0
4566 00008B8A 7232              <1>       jb     short cmd_help_retn
4567                            <1>
4568 00008B8C 56                <1>       push   esi
4569 00008B8D BE[1F4C0100]      <1>       mov    esi, nextline
4570 00008B92 E88EE8FFFF        <1>       call   print_msg
4571 00008B97 5E                <1>       pop    esi
4572 00008B98 EBE8              <1>       jmp    short cmd_help_next_w
4573                            <1>
4574                            <1> cmp_cmd_remark:
4575 00008B9A 3C2A              <1>       cmp    al, '*'
4576 00008B9C 0F8595020000      <1>         jne    cmp_cmd_external
4577 00008BA2 46                <1>       inc    esi
4578 00008BA3 BF[188A0100]      <1>       mov    edi, Remark
4579 00008BA8 8A06              <1>       mov    al, [esi]
4580 00008BAA 3C20              <1>       cmp    al, 20h
4581 00008BAC 7707              <1>       ja     short cmd_remark_write
4582 00008BAE 89FE              <1>       mov    esi, edi ; Remark
4583 00008BB0 E970E8FFFF        <1>       jmp    print_msg
4584                            <1>
4585                            <1> cmd_remark_write:
4586 00008BB5 AA                <1>       stosb
4587 00008BB6 AC                <1>       lodsb
4588 00008BB7 3C20              <1>       cmp    al, 20h
4589 00008BB9 73FA              <1>       jnb    short cmd_remark_write
4590 00008BBB C60700            <1>       mov    byte [edi], 0
4591                            <1>
4592                            <1> cmd_help_retn:
4593                            <1> cmd_remark_retn:
4594                            <1> cd_retn:
4595 00008BBE C3                <1>       retn
4596                            <1>
4597                            <1> c_2:
4598 00008BBF 80F902            <1>       cmp    cl, 2
4599 00008BC2 0F87AF000000      <1>         ja     c_3
4600 00008BC8 BE[668A0100]      <1>       mov    esi, CommandBuffer
4601 00008BCD 72A9              <1>       jb     short c_1
4602                            <1>
4603                            <1> cmp_cmd_cd:
4604 00008BCF 66AD              <1>       lodsw
4605 00008BD1 663D4344          <1>       cmp    ax, 'CD'
4606 00008BD5 7551              <1>       jne    short cmp_cmd_drive
4607 00008BD7 46                <1>         inc  esi
4608                            <1> cd_0:
4609 00008BD8 668B06            <1>       mov    ax, [esi]
4610 00008BDB 3C20              <1>       cmp    al, 20h
4611 00008BDD 76DF              <1>       jna    short cd_retn
4612                            <1>       ; 10/02/2016
4613 00008BDF 80FC3A            <1>       cmp    ah, ':'
```

```
4614 00008BE2 7504              <1>       jne    short cd_1
4615 00008BE4 46                <1>       inc    esi
4616 00008BE5 46                <1>       inc    esi
4617 00008BE6 EB49              <1>       jmp    short cd_2
4618                            <1>
4619                            <1> cd_1: ; change current directory
4620                            <1>       ; 29/11/2009
4621                            <1>       ; AH = CDh   ; to separate 'CD' command from others
4622                            <1>                    ; for restoring current directory
4623                            <1>                    ; 0CDh sign is for saving cdir into
4624                            <1>                    ; DOS drv description table cdir area
4625                            <1>
4626 00008BE8 B4CD              <1>       mov    ah, 0CDh ; mov byte [CD_COMMAND], 0CDh
4627                            <1>
4628 00008BEA E81D230000        <1>       call   change_current_directory
4629 00008BEF 0F8337220000      <1>         jnc     change_prompt_dir_string
4630                            <1>
4631                            <1> cd_error_messages:
4632 00008BF5 3C03              <1>       cmp    al, 3
4633 00008BF7 740C              <1>       je     short cd_path_not_found
4634                            <1>       ; 16/10/2016 (15h -> 15)
4635 00008BF9 3C0F              <1>       cmp    al, 15 ; drive not ready error
4636 00008BFB 7459              <1>       je     short cd_drive_not_ready
4637 00008BFD 3C11              <1>       cmp    al, 17 ; read error
4638 00008BFF 7455              <1>       je     short cd_drive_not_ready
4639 00008C01 3C13              <1>       cmp    al, 19 ; ; Bad directory/path name
4640 00008C03 7466              <1>       je     short cd_command_failed
4641                            <1>
4642                            <1> cd_path_not_found:
4643 00008C05 50                <1>       push   eax ; 29/12/2017
4644                            <1>       ;push  ax
4645 00008C06 BE[85420100]      <1>       mov    esi, Msg_Dir_Not_Found
4646 00008C0B E815E8FFFF        <1>       call   print_msg
4647                            <1>       ;pop   ax
4648 00008C10 58                <1>       pop    eax ; 29/12/2017
4649 00008C11 3A25[B4890100]    <1>       cmp    ah, [Current_Dir_Level]
4650 00008C17 0F830F220000      <1>         jnb     change_prompt_dir_string
4651 00008C1D 8825[B4890100]    <1>       mov    [Current_Dir_Level], ah
4652 00008C23 E904220000        <1>         jmp     change_prompt_dir_string
4653                            <1>
4654                            <1> cmp_cmd_drive: ; change current drive
4655                            <1>       ; C:, D:, E: etc.
4656 00008C28 80FC3A            <1>       cmp    ah, ':'
4657 00008C2B 0F8506020000      <1>         jne     cmp_cmd_external
4658                            <1>
4659                            <1> cd_2: ; 'CD C:', 'CD D:' ...
4660 00008C31 803E20            <1>       cmp    byte [esi], 20h
4661 00008C34 0F8707020000      <1>         ja      loc_cmd_failed
4662                            <1>
4663 00008C3A 24DF              <1>       and    al, 0DFh
4664 00008C3C 2C41              <1>       sub    al, 'A'
4665 00008C3E 0F82FD010000      <1>         jc      loc_cmd_failed
4666                            <1>
4667 00008C44 3A05[823F0100]    <1>         cmp     al, [Last_DOS_DiskNo]
4668 00008C4A 770A              <1>         ja   short cd_drive_not_ready
4669                            <1>
4670 00008C4C 88C2              <1>       mov    dl, al
4671 00008C4E E85BF3FFFF        <1>       call   change_current_drive
4672 00008C53 7201              <1>       jc     short cd_drive_not_ready
4673 00008C55 C3                <1>       retn
4674                            <1>
4675                            <1> cd_drive_not_ready:
4676 00008C56 BE[42420100]      <1>       mov    esi, Msg_Not_Ready_Read_Err
4677 00008C5B E8C5E7FFFF        <1>       call   print_msg
4678                            <1>
4679                            <1> cd_fail_drive_restart:
4680 00008C60 8A15[B6890100]    <1>       mov    dl, [Current_Drv]
4681                            <1>       ;call  change_current_drive
4682 00008C66 E943F3FFFF        <1>         jmp     change_current_drive
4683                            <1>       ;retn
4684                            <1>
4685                            <1> cd_command_failed:
4686 00008C6B BE[23420100]      <1>       mov    esi, Msg_Bad_Command
4687 00008C70 E8B0E7FFFF        <1>       call   print_msg
4688 00008C75 EBE9              <1>       jmp    short cd_fail_drive_restart
4689                            <1>
4690                            <1> c_3:
4691                            <1> cmp_cmd_dir:
4692 00008C77 BF[E23F0100]      <1>       mov    edi, Cmd_Dir
4693 00008C7C E8D3010000        <1>       call   cmp_cmd
4694 00008C81 0F8380020000      <1>       jnc    print_directory_list
4695                            <1>
4696                            <1> cmp_cmd_cls:
4697 00008C87 B103              <1>       mov    cl, 3
4698 00008C89 BF[1E400100]      <1>       mov    edi, Cmd_Cls
4699 00008C8E E8C1010000        <1>       call   cmp_cmd
4700 00008C93 0F83A2E7FFFF      <1>         jnc  clear_screen
4701                            <1>
4702                            <1> cmp_cmd_ver:
4703 00008C99 B103              <1>       mov    cl, 3
4704 00008C9B BF[EC3F0100]      <1>       mov    edi, Cmd_Ver
4705 00008CA0 E8AF010000        <1>       call   cmp_cmd
4706 00008CA5 720A              <1>       jc     short cmp_cmd_mem
4707                            <1>
4708 00008CA7 BE[8A3F0100]      <1>       mov    esi, mainprog_Version
4709                            <1>       ;call  print_msg
4710 00008CAC E974E7FFFF        <1>       jmp    print_msg
4711                            <1>       ;retn
4712                            <1>
4713                            <1> cmp_cmd_mem:
4714 00008CB1 B103              <1>       mov    cl, 3
4715 00008CB3 BF[54400100]      <1>       mov    edi, Cmd_Mem
4716 00008CB8 E897010000        <1>       call   cmp_cmd
4717 00008CBD 0F83DFB4FFFF      <1>       jnc    memory_info
4718                            <1>
```

```
4719                                    <1> cmp_cmd_del:
4720 00008CC3 B103                      <1>         mov     cl, 3
4721 00008CC5 BF[27400100]              <1>         mov     edi, Cmd_Del
4722 00008CCA E885010000                <1>         call    cmp_cmd
4723 00008CCF 0F83280F0000              <1>           jnc     delete_file
4724                                    <1>
4725                                    <1> cmp_cmd_set:
4726 00008CD5 B103                      <1>         mov     cl, 3
4727 00008CD7 BF[1A400100]              <1>         mov     edi, Cmd_Set
4728 00008CDC E873010000                <1>         call    cmp_cmd
4729 00008CE1 0F83C9170000              <1>           jnc     set_get_env
4730                                    <1>
4731                                    <1> cmp_cmd_run:
4732 00008CE7 B103                      <1>         mov     cl, 3
4733 00008CE9 BF[16400100]              <1>         mov     edi, Cmd_Run
4734 00008CEE E861010000                <1>         call    cmp_cmd
4735                                    <1>         ; 07/05/2016
4736 00008CF3 0F823E010000              <1>           jc      cmp_cmd_external
4737 00008CF9 E90F1E0000                <1>         jmp     load_and_execute_file
4738                                    <1> c_5:
4739                                    <1> cmp_cmd_mkdir:
4740 00008CFE BF[3F400100]              <1>         mov     edi, Cmd_Mkdir
4741 00008D03 E84C010000                <1>         call    cmp_cmd
4742 00008D08 0F83990A0000              <1>           jnc     make_directory
4743                                    <1>
4744                                    <1> cmp_cmd_rmdir:
4745 00008D0E B105                      <1>         mov     cl, 5
4746 00008D10 BF[39400100]              <1>         mov     edi, Cmd_Rmdir
4747 00008D15 E83A010000                <1>         call    cmp_cmd
4748 00008D1A 0F83AA0B0000              <1>           jnc     delete_directory
4749                                    <1>
4750                                    <1> cmp_cmd_chdir:
4751 00008D20 B105                      <1>         mov     cl, 5
4752 00008D22 BF[76400100]              <1>         mov     edi, Cmd_Chdir
4753 00008D27 E828010000                <1>         call    cmp_cmd
4754 00008D2C 0F8205010000              <1>           jc      cmp_cmd_external
4755                                    <1>
4756 00008D32 E9A1FEFFFF                <1>         jmp     cd_0
4757                                    <1>
4758                                    <1> c_6:
4759 00008D37 80F906                    <1>         cmp     cl, 6
4760 00008D3A 0F87E0000000              <1>           ja      c_8
4761 00008D40 72BC                      <1>         jb      short c_5
4762                                    <1> cmp_cmd_prompt:
4763 00008D42 BF[F53F0100]              <1>         mov     edi, Cmd_Prompt
4764 00008D47 E808010000                <1>         call    cmp_cmd
4765 00008D4C 722F                      <1>           jc    short cmp_cmd_volume
4766                                    <1> get_prompt_name_fchar:
4767 00008D4E AC                        <1>         lodsb
4768 00008D4F 3C20                      <1>         cmp     al, 20h
4769 00008D51 74FB                      <1>         je      short get_prompt_name_fchar
4770 00008D53 7713                      <1>         ja      short loc_change_prompt_label
4771                                    <1> default_command_prompt: ; 31/12/2017 ('sysprompt')
4772 00008D55 BE[D63F0100]              <1>         mov     esi, TRDOSPromptLabel
4773 00008D5A C7065452444F              <1>         mov     dword [esi], "TRDO"
4774 00008D60 66C746045300              <1>                 mov     word [esi+4], "S"
4775                                    <1> loc_cmd_prompt_return:
4776 00008D66 C3                        <1>         retn
4777                                    <1>
4778                                    <1> set_command_prompt: ; 31/12/2017 ('sysprompt')
4779 00008D67 AC                        <1>         lodsb
4780                                    <1> loc_change_prompt_label:
4781 00008D68 66B90B00                  <1>         mov     cx, 11
4782 00008D6C BF[D63F0100]              <1>         mov     edi, TRDOSPromptLabel
4783                                    <1> put_char_new_prompt_label:
4784 00008D71 AA                        <1>         stosb
4785 00008D72 AC                        <1>         lodsb
4786 00008D73 3C20                      <1>         cmp     al, 20h
4787 00008D75 7202                      <1>         jb      short pass_put_new_prompt_label
4788 00008D77 E2F8                      <1>         loop    put_char_new_prompt_label
4789                                    <1> pass_put_new_prompt_label:
4790 00008D79 C60700                    <1>         mov     byte [edi], 0
4791 00008D7C C3                        <1>         retn
4792                                    <1>
4793                                    <1> cmp_cmd_volume:
4794 00008D7D B106                      <1>         mov     cl, 6
4795 00008D7F BF[FC3F0100]              <1>         mov     edi, Cmd_Volume
4796 00008D84 E8CB000000                <1>         call    cmp_cmd
4797 00008D89 7255                      <1>           jc    short cmp_cmd_attrib
4798                                    <1>
4799                                    <1> cmd_vol1:
4800 00008D8B AC                        <1>         lodsb
4801 00008D8C 3C20                      <1>         cmp     al, 20h
4802 00008D8E 7707                      <1>         ja      short cmd_vol2
4803 00008D90 A0[B6890100]              <1>         mov     al, [Current_Drv]
4804 00008D95 EB3D                      <1>         jmp     short cmd_vol4
4805                                    <1> cmd_vol2:
4806 00008D97 3C41                      <1>         cmp     al, 'A'
4807 00008D99 0F82A2000000              <1>           jb      loc_cmd_failed
4808 00008D9F 3C7A                      <1>         cmp     al, 'z'
4809 00008DA1 0F879A000000              <1>           ja      loc_cmd_failed
4810 00008DA7 3C5A                      <1>         cmp     al, 'Z'
4811 00008DA9 760A                      <1>         jna     short cmd_vol3
4812 00008DAB 3C61                      <1>         cmp     al, 'a'
4813 00008DAD 0F828E000000              <1>           jb      loc_cmd_failed
4814 00008DB3 24DF                      <1>         and     al, 0DFh
4815                                    <1> cmd_vol3:
4816 00008DB5 8A26                      <1>         mov     ah, [esi]
4817 00008DB7 80FC3A                    <1>         cmp     ah, ':'
4818 00008DBA 0F8581000000              <1>           jne     loc_cmd_failed
4819 00008DC0 2C41                      <1>         sub     al, 'A'
4820 00008DC2 3A05[823F0100]            <1>           cmp     al, [Last_DOS_DiskNo]
4821 00008DC8 760A                      <1>         jna     short cmd_vol4
4822                                    <1>
4823 00008DCA BE[42420100]              <1>         mov     esi, Msg_Not_Ready_Read_Err
```

```
4824 00008DCF E951E6FFFF          <1>       jmp    print_msg
4825                              <1>
4826                              <1> cmd_vol4:
4827 00008DD4 E88EFAFFFF          <1>       call   print_volume_info
4828 00008DD9 0F8277FEFFFF        <1>         jc     cd_drive_not_ready
4829 00008DDF C3                  <1>       retn
4830                              <1>
4831                              <1> cmp_cmd_attrib:
4832 00008DE0 B106                <1>       mov    cl, 6
4833 00008DE2 BF[2B400100]        <1>       mov    edi, Cmd_Attrib
4834 00008DE7 E868000000          <1>       call   cmp_cmd
4835 00008DEC 0F831D0F0000        <1>         jnc    set_file_attributes
4836                              <1>
4837                              <1> cmp_cmd_rename:
4838 00008DF2 B106                <1>       mov    cl, 6
4839 00008DF4 BF[32400100]        <1>       mov    edi, Cmd_Rename
4840 00008DF9 E856000000          <1>       call   cmp_cmd
4841 00008DFE 0F8353110000        <1>         jnc    rename_file
4842                              <1>
4843                              <1> cmp_cmd_device:
4844 00008E04 B106                <1>       mov    cl, 6
4845 00008E06 BF[67400100]        <1>       mov    edi, Cmd_Device
4846 00008E0B E844000000          <1>       call   cmp_cmd
4847 00008E10 7225                <1>         jc   short cmp_cmd_external
4848                              <1>
4849 00008E12 C3                  <1>       retn
4850                              <1>
4851                              <1> c_7:
4852                              <1> cmp_cmd_devlist:
4853 00008E13 BF[6E400100]        <1>       mov    edi, Cmd_DevList
4854 00008E18 E837000000          <1>       call   cmp_cmd
4855 00008E1D 7218                <1>         jc   short cmp_cmd_external
4856                              <1>
4857                              <1> loc_cmd_return:
4858 00008E1F C3                  <1>       retn
4859                              <1>
4860                              <1> c_8:
4861 00008E20 80F908              <1>         cmp  cl, 8
4862 00008E23 7712                <1>       ja     short cmp_cmd_external
4863 00008E25 72EC                <1>       jb     short c_7
4864                              <1>
4865                              <1> cmp_cmd_longname:
4866 00008E27 BF[03400100]        <1>       mov    edi, Cmd_LongName
4867 00008E2C E823000000          <1>       call   cmp_cmd
4868 00008E31 0F8350060000        <1>         jnc    get_and_print_longname
4869                              <1>
4870                              <1> cmp_cmd_external:
4871                              <1>       ; 07/05/2016
4872                              <1>       ; 22/04/2016
4873 00008E37 BE[668A0100]        <1>       mov    esi, CommandBuffer
4874 00008E3C E9CC1C0000          <1>       jmp    loc_run_check_filename
4875                              <1>
4876                              <1> loc_cmd_failed:
4877 00008E41 803D[668A0100]20    <1>       cmp    byte [CommandBuffer], 20h
4878 00008E48 76D5                <1>       jna    short loc_cmd_return
4879 00008E4A BE[23420100]        <1>       mov    esi, Msg_Bad_Command
4880                              <1> ;     call   print_msg
4881                              <1> ;loc_cmd_return:
4882                              <1> ;     retn
4883 00008E4F E9D1E5FFFF          <1>       jmp    print_msg
4884                              <1>
4885                              <1> cmp_cmd:
4886                              <1>        ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
4887 00008E54 BE[668A0100]        <1>         mov esi, CommandBuffer
4888                              <1>         ; edi = internal command word (ASCIIZ)
4889                              <1>        ; ecx = command length (<=8)
4890                              <1> cmp_cmd_1:
4891 00008E59 AC                  <1>       lodsb
4892 00008E5A AE                  <1>       scasb
4893 00008E5B 750D                <1>       jne    short cmp_cmd_3
4894 00008E5D E2FA                <1>       loop   cmp_cmd_1
4895 00008E5F AC                  <1>       lodsb
4896 00008E60 3C20                <1>       cmp    al, 20h
4897 00008E62 7703                <1>       ja     short cmp_cmd_2
4898 00008E64 30C0                <1>       xor    al, al
4899                              <1>       ; ZF = 1 -> internal command word matches
4900 00008E66 C3                  <1>       retn
4901                              <1> cmp_cmd_2:
4902                              <1>       ; ZF = 0 (CF = 0) -> external command word
4903 00008E67 58                  <1>       pop    eax ; no return to the caller from here
4904 00008E68 EBCD                <1>       jmp    cmp_cmd_external
4905                              <1> cmp_cmd_3:
4906 00008E6A F9                  <1>       stc
4907                              <1>       ; CF = 1 -> internal command word does not match
4908 00008E6B C3                  <1>       retn
4909                              <1>
4910                              <1> loc_run_cmd_failed:
4911                              <1>       ; 15/03/2016
4912                              <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
4913                              <1>       ; 07/12/2009 (CMD_INTR.ASM)
4914                              <1>       ; 29/11/2009
4915                              <1>
4916 00008E6C E863000000          <1>       call   restore_cdir_after_cmd_fail
4917                              <1>
4918                              <1> loc_run_cmd_failed_cmp_al:
4919                              <1>       ; End of Restore_CDIR code (29/11/2009)
4920                              <1>
4921 00008E71 3C01                <1>       cmp    al, 1 ; Bad command or file name
4922 00008E73 74CC                <1>       je     loc_cmd_failed
4923                              <1> loc_run_dir_not_found:
4924 00008E75 3C03                <1>       cmp    al, 3
4925 00008E77 750A                <1>       jne    short loc_run_file_notfound_msg
4926                              <1>       ; Path not found (MS-DOS Error Code = 3)
4927 00008E79 BE[85420100]        <1>       mov    esi, Msg_Dir_Not_Found
4928 00008E7E E9A2E5FFFF          <1>       jmp    print_msg
```

```
                                <1>
                                <1> loc_run_file_notfound_msg:
00008E83 3C02                   <1>       cmp    al, 2 ; File not found
00008E85 750A                   <1>       jne    short loc_run_file_drv_read_err
                                <1>
                                <1> loc_print_file_notfound_msg:
00008E87 BE[9C420100]           <1>         mov     esi, Msg_File_Not_Found
                                <1>       ;call  proc_printmsg
                                <1>       ;retn
00008E8C E994E5FFFF             <1>       jmp    print_msg
                                <1>
                                <1> loc_run_file_drv_read_err:
                                <1>       ; Err: 17 (Read fault)
00008E91 3C11                   <1>       cmp    al, 17 ; Drive not ready or read error
00008E93 7404                   <1>       je     short loc_run_file_print_drv_read_err
                                <1>       ;
00008E95 3C0F                   <1>       cmp    al, 15 ; Drive not ready (or read error)
00008E97 750A                   <1>       jne    short loc_run_file_toobig
                                <1>
                                <1> loc_run_file_print_drv_read_err:
00008E99 BE[42420100]           <1>       mov    esi, Msg_Not_Ready_Read_Err
00008E9E E982E5FFFF             <1>       jmp    print_msg
                                <1>
                                <1> loc_run_file_toobig:
00008EA3 3C08                   <1>       cmp    al, 8 ; Not enough free memory to load&run file
00008EA5 750A                   <1>       jne    short loc_run_file_perm_denied
00008EA7 BE[E7420100]           <1>       mov    esi, Msg_Insufficient_Memory
00008EAC E974E5FFFF             <1>       jmp    print_msg
                                <1>
                                <1> loc_run_file_perm_denied:
                                <1>       ; 29/12/2017
00008EB1 3C0B                   <1>       cmp    al, ERR_PERM_DENIED ; 11 ; Permission denied
00008EB3 750A                   <1>       jne    short loc_run_misc_error
00008EB5 BE[7C440100]           <1>       mov    esi, Msg_Permission_Denied
00008EBA E966E5FFFF             <1>       jmp    print_msg
                                <1>
                                <1>       ; 15/03/2016
                                <1> print_misc_error_msg:
                                <1> loc_run_misc_error:
                                <1>       ; AL = Error code
00008EBF E86CB3FFFF             <1>       call   bytetohex
00008EC4 66A3[1B430100]         <1>         mov     [error_code_hex], ax
                                <1>
00008ECA BE[FE420100]           <1>       mov    esi, Msg_Error_Code
                                <1>       ;call  print_msg
                                <1>       ;retn
                                <1>
00008ECF E951E5FFFF             <1>       jmp    print_msg
                                <1>
                                <1> restore_cdir_after_cmd_fail:
                                <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
00008ED4 50                     <1>       push   eax
00008ED5 8A3D[12910100]         <1>       mov    bh, [RUN_CDRV] ; it is set at the beginning
                                <1>                              ; of the 'run' command.
00008EDB 3A3D[B6890100]         <1>       cmp    bh, [Current_Drv]
00008EE1 7409                   <1>       je     short loc_run_restore_cdir
00008EE3 88FA                   <1>       mov    dl, bh
00008EE5 E8C4F0FFFF             <1>       call   change_current_drive
00008EEA EB19                   <1>       jmp    short loc_run_err_pass_restore_cdir
                                <1>
                                <1> loc_run_restore_cdir:
00008EEC 803D[833F0100]00       <1>       cmp    byte [Restore_CDIR], 0
00008EF3 7610                   <1>       jna    short loc_run_err_pass_restore_cdir
00008EF5 30DB                   <1>       xor    bl, bl
00008EF7 0FB7F3                 <1>       movzx  esi, bx
00008EFA 81C600010900           <1>       add    esi, Logical_DOSDisks
00008F00 E860F1FFFF             <1>       call   restore_current_directory
                                <1>
                                <1> loc_run_err_pass_restore_cdir:
00008F05 58                     <1>       pop    eax
00008F06 C3                     <1>       retn
                                <1>
                                <1> print_directory_list:
                                <1>       ; 10/02/2016
                                <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 06/12/2009 ('cmp_cmd_dir')
                                <1>       ;
00008F07 66C705[54920100]00-    <1>       mov    word [AttributesMask], 0800h ; ..except volume names..
00008F0F 08                     <1>
00008F10 A0[B6890100]           <1>       mov    al, [Current_Drv]
00008F15 A2[12910100]           <1>       mov    [RUN_CDRV], al
                                <1> get_dfname_fchar:
00008F1A AC                     <1>       lodsb
00008F1B 3C20                   <1>       cmp    al, 20h
00008F1D 74FB                   <1>       je     short get_dfname_fchar
00008F1F 0F82A4000000           <1>         jb      loc_print_dir_call_all
00008F25 3C2D                   <1>       cmp    al, '-'
00008F27 7542                   <1>       jne    short loc_print_dir_call_flt
                                <1> get_next_attr_char:
00008F29 AC                     <1>       lodsb
00008F2A 3C20                   <1>       cmp    al, 20h
00008F2C 74FB                   <1>       je     short get_next_attr_char
00008F2E 0F820DFFFFFF           <1>         jb      loc_cmd_failed
00008F34 24DF                   <1>       and    al, 0DFh
00008F36 3C44                   <1>       cmp    al, 'D' ; directories only ?
00008F38 7512                   <1>       jne    short pass_only_directories
00008F3A AC                     <1>       lodsb
00008F3B 3C20                   <1>       cmp    al, 20h
00008F3D 0F87FEFEFFFF           <1>         ja      loc_cmd_failed
00008F43 800D[54920100]10       <1>       or     byte [AttributesMask], 10h ; ..directory..
00008F4A EB18                   <1>       jmp    short get_dfname_fchar_attr
                                <1> pass_only_directories:
00008F4C 3C46                   <1>       cmp    al, 'F'      ; files only ?
00008F4E 0F85B0000000           <1>         jne     check_attr_s
00008F54 AC                     <1>       lodsb
```

```
5033 00008F55 3C20               <1>       cmp    al, 20h
5034 00008F57 0F87E4FEFFFF       <1>         ja      loc_cmd_failed
5035 00008F5D 800D[55920100]10   <1>       or     byte [AttributesMask+1], 10h ; ..except directories..
5036                             <1> get_dfname_fchar_attr:
5037 00008F64 AC                 <1>       lodsb
5038 00008F65 3C20               <1>       cmp    al, 20h
5039 00008F67 74FB               <1>       je     short get_dfname_fchar_attr
5040 00008F69 725E               <1>       jb     short loc_print_dir_call_all
5041                             <1>
5042                             <1> loc_print_dir_call_flt:
5043 00008F6B 4E                 <1>       dec    esi
5044 00008F6C BF[56920100]       <1>       mov    edi, FindFile_Drv
5045 00008F71 E8AC250000         <1>       call   parse_path_name
5046 00008F76 7308               <1>       jnc    short loc_print_dir_change_drv_1
5047 00008F78 3C01               <1>       cmp    al, 1
5048 00008F7A 0F87ECFEFFFF       <1>         ja      loc_run_cmd_failed
5049                             <1>
5050                             <1> loc_print_dir_change_drv_1:
5051 00008F80 8A15[56920100]     <1>       mov    dl, [FindFile_Drv]
5052                             <1> loc_print_dir_change_drv_2:
5053 00008F86 3A15[12910100]     <1>       cmp    dl, [RUN_CDRV]
5054 00008F8C 740B               <1>       je     short loc_print_dir_change_directory
5055 00008F8E E81BF0FFFF         <1>       call   change_current_drive
5056 00008F93 0F82D3FEFFFF       <1>         jc      loc_run_cmd_failed
5057                             <1> loc_print_dir_change_directory:
5058 00008F99 803D[57920100]20   <1>       cmp    byte [FindFile_Directory], 20h ; 0 or 20h ?
5059 00008FA0 761D               <1>       jna    short pass_print_dir_change_directory
5060                             <1>
5061 00008FA2 FE05[833F0100]     <1>       inc    byte [Restore_CDIR]
5062 00008FA8 BE[57920100]       <1>       mov    esi, FindFile_Directory
5063 00008FAD 30E4               <1>       xor    ah, ah ; CD_COMMAND sign -> 0
5064 00008FAF E8581F0000         <1>       call   change_current_directory
5065 00008FB4 0F82B2FEFFFF       <1>         jc      loc_run_cmd_failed
5066                             <1>
5067                             <1> loc_print_dir_change_prompt_dir_string:
5068 00008FBA E86D1E0000         <1>       call   change_prompt_dir_string
5069                             <1>
5070                             <1> pass_print_dir_change_directory:
5071 00008FBF BE[98920100]       <1>       mov    esi, FindFile_Name
5072 00008FC4 803E20             <1>       cmp    byte [esi], 20h ; ; 0 or 20h ?
5073 00008FC7 7706               <1>       ja     short loc_print_dir_call
5074                             <1>
5075                             <1> loc_print_dir_call_all:
5076 00008FC9 C7062A2E2A00       <1>       mov    dword [esi], '*.*'
5077                             <1> loc_print_dir_call:
5078 00008FCF E87E000000         <1>       call   print_directory
5079                             <1>
5080 00008FD4 8A15[12910100]     <1>       mov    dl, [RUN_CDRV]  ; it is set at the beginning
5081 00008FDA 3A15[B6890100]     <1>       cmp    dl, [Current_Drv]
5082 00008FE0 7406               <1>       je     short loc_print_dir_call_restore_cdir_retn
5083 00008FE2 E8C7EFFFFF         <1>       call   change_current_drive
5084 00008FE7 C3                 <1>       retn
5085                             <1>
5086                             <1> loc_print_dir_call_restore_cdir_retn:
5087 00008FE8 803D[833F0100]00   <1>       cmp    byte [Restore_CDIR], 0
5088 00008FEF 7610               <1>       jna    short pass_print_dir_call_restore_cdir_retn
5089                             <1>
5090 00008FF1 BE00010900         <1>       mov    esi, Logical_DOSDisks
5091 00008FF6 31C0               <1>       xor    eax, eax
5092 00008FF8 88D4               <1>       mov    ah, dl
5093 00008FFA 01C6               <1>       add    esi, eax
5094                             <1>
5095 00008FFC E864F0FFFF         <1>       call   restore_current_directory
5096                             <1>
5097                             <1> pass_print_dir_call_restore_cdir_retn:
5098 00009001 C3                 <1>       retn
5099                             <1>
5100                             <1> check_attr_s_cap:
5101 00009002 24DF               <1>       and    al, 0DFh
5102                             <1> check_attr_s:
5103 00009004 3C53               <1>       cmp    al, 'S'
5104 00009006 7514               <1>       jne    short pass_attr_s
5105 00009008 800D[54920100]04   <1>       or     byte [AttributesMask], 4 ; system
5106 0000900F AC                 <1>       lodsb
5107 00009010 3C20               <1>       cmp    al, 20h
5108 00009012 0F844CFFFFFF       <1>         je      get_dfname_fchar_attr
5109 00009018 72AF               <1>       jb     short loc_print_dir_call_all
5110 0000901A 24DF               <1>       and    al, 0DFh
5111                             <1> pass_attr_s:
5112 0000901C 3C48               <1>       cmp    al, 'H'
5113 0000901E 7514               <1>       jne    short pass_attr_h
5114 00009020 800D[54920100]02   <1>       or     byte [AttributesMask], 2 ; hidden
5115                             <1> pass_attr_shr:
5116 00009027 AC                 <1>       lodsb
5117 00009028 3C20               <1>       cmp    al, 20h
5118 0000902A 0F8434FFFFFF       <1>         je      get_dfname_fchar_attr
5119 00009030 7297               <1>       jb     short loc_print_dir_call_all
5120 00009032 EBCE               <1>       jmp    short check_attr_s_cap
5121                             <1>
5122                             <1> pass_attr_h:
5123 00009034 3C52               <1>       cmp    al, 'R'
5124 00009036 7509               <1>       jne    short pass_attr_r
5125 00009038 800D[54920100]01   <1>       or     byte [AttributesMask], 1 ; read only
5126 0000903F EBE6               <1>       jmp    short pass_attr_shr
5127                             <1>
5128                             <1> pass_attr_r:
5129 00009041 3C41               <1>       cmp    al, 'A'
5130 00009043 0F85F8FDFFFF       <1>         jne     loc_cmd_failed
5131 00009049 800D[54920100]20   <1>       or     byte [AttributesMask], 20h ; archive
5132 00009050 EBD5               <1>       jmp    short pass_attr_shr
5133                             <1>
5134                             <1> print_directory:
5135                             <1>       ; 13/05/2016
5136                             <1>       ; 11/02/2016
5137                             <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>
00009052 56                     <1>      push   esi
                                <1>
00009053 29C0                   <1>      sub    eax, eax
                                <1>
00009055 66A3[E0920100]         <1>      mov    word [Dir_Count], ax ; 0
0000905B 66A3[DE920100]         <1>      mov    word [File_Count], ax ; 0
00009061 A3[E2920100]           <1>      mov    dword [Total_FSize], eax ; 0
                                <1>
00009066 E8D0E3FFFF             <1>      call    clear_screen
                                <1>
0000906B 31C9                   <1>      xor    ecx, ecx
0000906D 8A2D[B6890100]         <1>      mov     ch, [Current_Drv] ; DirBuff_Drv - 'A'
00009073 A0[B7890100]           <1>      mov     al, [Current_Dir_Drv]
00009078 A2[40410100]           <1>      mov     [Dir_Drive_Name], al
0000907D BE00010900             <1>      mov    esi, Logical_DOSDisks
00009082 01CE                   <1>      add    esi, ecx
                                <1>
00009084 E858F9FFFF             <1>      call   move_volume_name_and_serial_no
00009089 730C                   <1>      jnc    short print_dir_strlen_check
                                <1>
0000908B 5E                     <1>      pop    esi
0000908C 8A3D[1E890100]         <1>      mov    bh, [ptty] ; [ACTIVE_PAGE]
                                <1>      ;call  beeper
                                <1>      ;retn
00009092 E9D592FFFF             <1>      jmp    beeper  ; beep ! and return
                                <1>
                                <1> print_dir_strlen_check:
00009097 BE[B9890100]           <1>      mov    esi, Current_Dir_Root
0000909C BF[DD410100]           <1>      mov    edi, Dir_Str_Root
                                <1>
                                <1>      ;xor   ecx, ecx
000090A1 8A0D[158A0100]         <1>        mov     cl, [Current_Dir_StrLen]
000090A7 FEC1                   <1>      inc    cl
000090A9 80F940                 <1>      cmp    cl, 64
000090AC 760D                   <1>      jna    short pass_print_dir_strlen_shorting
000090AE 46                     <1>      inc    esi
000090AF 01CE                   <1>      add    esi, ecx
000090B1 83EE40                 <1>      sub    esi, 64
000090B4 47                     <1>      inc    edi
000090B5 B82E2E2E20             <1>      mov    eax, '... '
000090BA AB                     <1>      stosd
                                <1>
                                <1> pass_print_dir_strlen_shorting:
000090BB F3A4                   <1>      rep    movsb
                                <1>
000090BD BE[33410100]           <1>      mov    esi, Dir_Drive_Str
000090C2 E85EE3FFFF             <1>      call   print_msg
                                <1>
000090C7 BE[92410100]           <1>      mov    esi, Vol_Serial_Header
000090CC E854E3FFFF             <1>      call   print_msg
                                <1>
000090D1 BE[D2410100]           <1>      mov    esi, Dir_Str_Header
000090D6 E84AE3FFFF             <1>      call   print_msg
                                <1>
000090DB BE[1D4C0100]           <1>      mov    esi, next2line
000090E0 E840E3FFFF             <1>      call   print_msg
                                <1>
                                <1> loc_print_dir_first_file:
000090E5 C605[F5920100]10       <1>      mov    byte [PrintDir_RowCounter], 16
000090EC 66A1[54920100]         <1>      mov    ax, [AttributesMask]
000090F2 5E                     <1>      pop    esi
                                <1>
000090F3 E859020000             <1>      call   find_first_file
000090F8 0F826F010000           <1>        jc     loc_dir_ok
                                <1>
                                <1> loc_dfname_use_this:
                                <1>      ; bl = File Attributes (bh = Long Name Entry Length)
000090FE F6C310                 <1>      test   bl, 10h  ; Is it a directory?
00009101 741B                   <1>      jz     short loc_not_dir
                                <1>
00009103 66FF05[E0920100]       <1>      inc    word [Dir_Count]
0000910A 89F2                   <1>      mov    edx, esi     ; FindFile_DirEntry address
0000910C BE[22430100]           <1>      mov    esi, Type_Dir; '<DIR>     '
00009111 BF[39430100]           <1>      mov    edi, Dir_Or_FileSize
                                <1>      ; move 10 bytes
00009116 A5                     <1>      movsd
00009117 A5                     <1>      movsd
00009118 66A5                   <1>      movsw
0000911A 89D6                   <1>      mov    esi, edx
0000911C EB36                   <1>      jmp     short loc_dir_attribute
                                <1>
                                <1> loc_not_dir:
0000911E 66FF05[DE920100]       <1>      inc    word [File_Count]
00009125 0105[E2920100]         <1>      add    [Total_FSize], eax
                                <1>
0000912B B90A000000             <1>      mov    ecx, 10  ; 32 bit divisor
00009130 89CF                   <1>      mov    edi, ecx
00009132 81C7[39430100]         <1>      add    edi, Dir_Or_FileSize
                                <1> loc_dir_rdivide:
00009138 29D2                   <1>      sub    edx, edx
0000913A F7F1                   <1>      div    ecx     ; remainder in dl (< 10)
0000913C 80C230                 <1>      add     dl, '0'      ; to make visible (ascii)
0000913F 4F                     <1>      dec    edi
00009140 8817                   <1>      mov     [edi], dl
00009142 21C0                   <1>      and    eax, eax
00009144 75F2                   <1>      jnz    short loc_dir_rdivide
                                <1>
                                <1> loc_dir_fill_space:
```

```
5243 00009146 81FF[39430100]      <1>       cmp     edi, Dir_Or_FileSize
5244 0000914C 7606                <1>       jna     short loc_dir_attribute
5245 0000914E 4F                  <1>       dec     edi
5246 0000914F C60720              <1>       mov     byte [edi], 20h
5247 00009152 EBF2                <1>       jmp     short loc_dir_fill_space
5248                              <1>
5249                              <1> loc_dir_attribute:
5250 00009154 C705[44430100]2020- <1>       mov   dword [File_Attribute], 20202020h
5250 0000915C 2020                <1>
5251                              <1>
5252 0000915E 80FB20              <1>       cmp   bl, 20h  ; Is it an archive file?
5253 00009161 7207                <1>       jb    short loc_dir_pass_arch
5254 00009163 C605[47430100]41    <1>       mov   byte [File_Attribute+3], 'A'
5255                              <1>
5256                              <1> loc_dir_pass_arch:
5257 0000916A 80E307              <1>       and   bl, 7
5258 0000916D 7428                <1>       jz    short loc_dir_file_name
5259 0000916F 88DF                <1>       mov   bh, bl
5260 00009171 80E303              <1>       and   bl, 3
5261 00009174 38DF                <1>       cmp   bh, bl
5262 00009176 7607                <1>       jna   short loc_dir_pass_s
5263 00009178 C605[44430100]53    <1>       mov   byte [File_Attribute], 'S'
5264                              <1>
5265                              <1> loc_dir_pass_s:
5266 0000917F 80E302              <1>       and     bl,2
5267 00009182 7407                <1>       jz      short loc_dir_pass_h
5268 00009184 C605[45430100]48    <1>       mov     byte [File_Attribute+1], 'H'
5269                              <1> loc_dir_pass_h:
5270 0000918B 80E701              <1>       and     bh,1
5271 0000918E 7407                <1>       jz      short loc_dir_file_name
5272 00009190 C605[46430100]52    <1>       mov     byte [File_Attribute+2], 'R'
5273                              <1> loc_dir_file_name:
5274                              <1>       ;mov     bx, [esi+18h] ; Date
5275                              <1>       ;mov     dx, [esi+16h] ; Time
5276 00009197 8B5E16              <1>       mov   ebx, [esi+16h]
5277 0000919A 89F1                <1>       mov   ecx, esi ; FindFile_DirEntry address
5278 0000919C BF[2C430100]        <1>       mov     edi, File_Name
5279                              <1>       ; move 8 bytes
5280 000091A1 A5                  <1>       movsd
5281 000091A2 A5                  <1>       movsd
5282 000091A3 C60720              <1>       mov   byte [edi], 20h
5283 000091A6 47                  <1>       inc   edi
5284                              <1>       ; move 3 bytes
5285 000091A7 66A5                <1>       movsw
5286 000091A9 A4                  <1>       movsb
5287 000091AA 89CE                <1>       mov   esi, ecx
5288                              <1>
5289                              <1> Dir_Time_start:
5290                              <1>       ;mov  ax, dx       ; Time
5291 000091AC 6689D8              <1>       mov   ax, bx
5292 000091AF 66C1E805            <1>       shr   ax, 5        ; shift right 5 times
5293 000091B3 6683E03F            <1>       and   ax, 0000111111b     ; Minute Mask
5294 000091B7 D40A                <1>       aam                ; Q([AL]/10)->AH
5295                              <1>                          ; R([AL]/10)->AL
5296                              <1>                          ; [AL]+[AH]= Minute as BCD
5297 000091B9 660D3030            <1>       or    ax, '00'     ; Convert to ASCII
5298 000091BD 86E0                <1>       xchg  ah, al
5299 000091BF 66A3[57430100]      <1>       mov   [File_Minute], ax
5300                              <1>
5301                              <1>       ;mov  al, dh
5302 000091C5 88F8                <1>       mov   al, bh
5303 000091C7 C0E803              <1>       shr   al, 3        ; shift right 3 times
5304 000091CA D40A                <1>       aam                ; [AL]+[AH]= Hours as BCD
5305 000091CC 660D3030            <1>       or    ax, '00'
5306 000091D0 86E0                <1>       xchg  ah, al
5307 000091D2 66A3[54430100]      <1>       mov     [File_Hour], ax
5308                              <1>
5309 000091D8 C1EB10              <1>       shr   ebx, 16            ; BX = Date
5310                              <1>
5311                              <1> Dir_Date_start:
5312 000091DB 6689D8              <1>       mov   ax, bx       ; Date
5313 000091DE 6683E01F            <1>       and   ax, 00011111b; Day Mask
5314 000091E2 D40A                <1>       aam                ; Q([AL]/10)->AH
5315                              <1>                          ; R([AL]/10)->AL
5316                              <1>                          ; [AL]+[AH]= Day as BCD
5317 000091E4 660D3030            <1>       or    ax, '00'     ; Convert to ASCII
5318 000091E8 86C4                <1>       xchg  al, ah
5319                              <1>
5320 000091EA 66A3[49430100]      <1>       mov   [File_Day], ax
5321                              <1>
5322 000091F0 6689D8              <1>       mov   ax, bx
5323 000091F3 66C1E805            <1>       shr   ax, 5        ; shift right 5 times
5324 000091F7 6683E00F            <1>       and   ax, 00001111b; Month Mask
5325 000091FB D40A                <1>       aam
5326 000091FD 660D3030            <1>       or    ax, '00'
5327 00009201 86E0                <1>       xchg  ah, al
5328 00009203 66A3[4C430100]      <1>       mov   [File_Month], ax
5329                              <1>
5330 00009209 6689D8              <1>       mov   ax, bx
5331 0000920C 66C1E809            <1>       shr     ax, 9
5332 00009210 6683E07F            <1>       and   ax, 01111111b; Result = Year - 1980
5333 00009214 6605BC07            <1>       add   ax, 1980
5334                              <1>
5335 00009218 B10A                <1>       mov   cl, 10
5336 0000921A F6F1                <1>       div   cl           ; Q -> AL, R -> AH
5337 0000921C 80CC30              <1>       or    ah, '0'
5338 0000921F 8825[52430100]      <1>       mov   [File_Year+3], ah
5339 00009225 D40A                <1>       aam
5340 00009227 86E0                <1>       xchg  ah, al
5341 00009229 80CC30              <1>       or    ah, '0'        ; Convert to ASCII
5342 0000922C 8825[51430100]      <1>       mov   [File_Year+2], ah
5343 00009232 D40A                <1>       aam
5344 00009234 86C4                <1>       xchg  al, ah
5345 00009236 660D3030            <1>       or    ax, '00'
5346 0000923A 66A3[4F430100]      <1>       mov   [File_Year], ax
```

```
                                <1>
                                <1> loc_show_line:
00009240 56                     <1>       push   esi
00009241 BE[2C430100]           <1>       mov     esi, File_Name
00009246 E8DAE1FFFF             <1>       call   print_msg
0000924B BE[1F4C0100]           <1>       mov    esi, nextline
00009250 E8D0E1FFFF             <1>       call   print_msg
00009255 5E                     <1>       pop    esi
                                <1>
00009256 FE0D[F5920100]         <1>       dec    byte [PrintDir_RowCounter]
0000925C 0F84D4000000           <1>         jz      pause_dir_scroll
                                <1>
                                <1> loc_next_entry:
00009262 E899010000             <1>       call   find_next_file
00009267 0F8391FEFFFF           <1>         jnc     loc_dfname_use_this
                                <1>
                                <1> loc_dir_ok:
0000926D B90A000000             <1>       mov     ecx, 10
00009272 66A1[E0920100]         <1>       mov    ax, [Dir_Count]
00009278 BF[6D430100]           <1>       mov    edi, Decimal_Dir_Count
0000927D 6639C8                 <1>       cmp    ax, cx ; 10
00009280 7216                   <1>       jb     short pass_ddc
00009282 47                     <1>       inc    edi
00009283 6683F864               <1>       cmp    ax, 100
00009287 720F                   <1>       jb     short pass_ddc
00009289 47                     <1>       inc    edi
0000928A 663DE803               <1>       cmp    ax, 1000
0000928E 7208                   <1>       jb     short pass_ddc
00009290 47                     <1>       inc    edi
00009291 663D1027               <1>       cmp    ax, 10000
00009295 7201                   <1>       jb     short pass_ddc
00009297 47                     <1>       inc    edi
                                <1> pass_ddc:
00009298 886F01                 <1>       mov     [edi+1], ch ; 0
                                <1> loc_ddc_rediv:
0000929B 31D2                   <1>       xor     edx, edx
0000929D 66F7F1                 <1>       div     cx   ; 10
000092A0 80C230                 <1>       add     dl, '0'
000092A3 8817                   <1>       mov     [edi], dl
000092A5 4F                     <1>       dec     edi
000092A6 6609C0                 <1>       or     ax, ax
000092A9 75F0                   <1>       jnz    short loc_ddc_rediv
                                <1>
000092AB 66A1[DE920100]         <1>       mov     ax, [File_Count]
000092B1 BF[5C430100]           <1>       mov     edi, Decimal_File_Count
000092B6 6639C8                 <1>       cmp     ax, cx ; 10
000092B9 7216                   <1>       jb      short pass_dfc
000092BB 47                     <1>       inc     edi
000092BC 6683F864               <1>       cmp     ax, 100
000092C0 720F                   <1>       jb      short pass_dfc
000092C2 47                     <1>       inc     edi
000092C3 663DE803               <1>       cmp     ax, 1000
000092C7 7208                   <1>       jb      short pass_dfc
000092C9 47                     <1>       inc     edi
000092CA 663D1027               <1>       cmp     ax, 10000
000092CE 7201                   <1>       jb      short pass_dfc
000092D0 47                     <1>       inc     edi
                                <1> pass_dfc:
                                <1>       ;mov    cx, 10
000092D1 886F01                 <1>       mov     [edi+1], ch ; 00
                                <1> loc_dfc_rediv:
                                <1>       ;xor   dx, dx
000092D4 30D2                   <1>       xor    dl, dl
000092D6 66F7F1                 <1>       div    cx
000092D9 80C230                 <1>       add    dl, '0'
000092DC 8817                   <1>       mov    [edi], dl
000092DE 4F                     <1>       dec    edi
000092DF 6609C0                 <1>       or     ax, ax
000092E2 75F0                   <1>       jnz    short loc_dfc_rediv
                                <1>
000092E4 BF[F4920100]           <1>       mov     edi, TFS_Dec_End
                                <1>         ;mov    byte [edi], 0
000092E9 A1[E2920100]           <1>       mov     eax, [Total_FSize]
                                <1>       ;mov    ecx, 10
                                <1> rediv_tfs_hex:
                                <1>       ;sub   edx, edx
000092EE 28D2                   <1>       sub    dl, dl
000092F0 F7F1                   <1>       div    ecx
000092F2 80C230                 <1>       add    dl, '0'
000092F5 4F                     <1>       dec     edi
000092F6 8817                   <1>       mov     [edi], dl
000092F8 21C0                   <1>       and    eax, eax
000092FA 75F2                   <1>       jnz    short rediv_tfs_hex
                                <1>
000092FC 893D[E6920100]         <1>       mov    [TFS_Dec_Begin], edi
00009302 BE[5A430100]           <1>       mov    esi, Decimal_File_Count_Header
00009307 E819E1FFFF             <1>       call   print_msg
0000930C BE[62430100]           <1>       mov    esi, str_files
00009311 E80FE1FFFF             <1>       call   print_msg
00009316 BE[73430100]           <1>       mov    esi, str_dirs
0000931B E805E1FFFF             <1>       call   print_msg
00009320 8B35[E6920100]         <1>       mov    esi, [TFS_Dec_Begin]
00009326 E8FAE0FFFF             <1>       call   print_msg
0000932B BE[84430100]           <1>       mov    esi, str_bytes
00009330 E8F0E0FFFF             <1>       call   print_msg
                                <1>
00009335 C3                     <1>       retn
                                <1>
                                <1> pause_dir_scroll:
00009336 28E4                   <1>       sub    ah, ah
00009338 E8A67BFFFF             <1>       call   int16h
0000933D 3C1B                   <1>       cmp    al, 1Bh
0000933F 0F8428FFFFFF           <1>         je      loc_dir_ok
00009345 C605[F5920100]10       <1>       mov    byte [PrintDir_RowCounter], 16 ; Reset counter
0000934C E911FFFFFF             <1>         jmp     loc_next_entry
```

```
                                <1>
                                <1> find_first_file:
                                <1>       ; 11/02/2016
                                <1>       ; 10/02/2016
                                <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 09/10/2011
                                <1>       ; 17/09/2009
                                <1>       ; 2005
                                <1>       ; INPUT ->
                                <1>       ;     ESI = ASCIIZ File/Dir Name Address (in Current Directory)
                                <1>       ;     AL = Attributes AND mask (The AND result must be equal to AL)
                                <1>       ;          bit 0 = Read Only
                                <1>       ;          bir 1 = Hidden
                                <1>       ;          bit 2 = System
                                <1>       ;          bit 3 = Volume Label
                                <1>       ;          bit 4 = Directory
                                <1>       ;          bit 5 = Archive
                                <1>       ;          bit 6 = Reserved, must be 0
                                <1>       ;          bit 7 = Reserved, must be 0
                                <1>       ;       AH = Attributes Negative AND mask (The AND result must be ZERO)
                                <1>       ;
                                <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>
00009351 66A3[A6920100]         <1>       mov   [FindFile_AttributesMask], ax
00009357 BF[A8920100]           <1>       mov   edi, FindFile_DirEntry ; TR-DOS Fullfilename formatted buffer
0000935C 31C0                   <1>       xor   eax, eax
0000935E B90B000000             <1>       mov   ecx, 11
00009363 F3AB                   <1>       rep   stosd  ; 44 bytes
                                <1>       ;stosw       ; +2 bytes
                                <1>
00009365 BF[98920100]           <1>       mov   edi, FindFile_Name ; FFF structure, offset 66
0000936A 39FE                   <1>       cmp   esi, edi
0000936C 7408                   <1>       je    short loc_fff_mfn_ok
0000936E 89FA                   <1>       mov   edx, edi
                                <1>       ; move 13 bytes
00009370 A5                     <1>       movsd
00009371 A5                     <1>       movsd
00009372 A5                     <1>       movsd
00009373 AA                     <1>       stosb
00009374 89D6                   <1>       mov   esi, edx
                                <1> loc_fff_mfn_ok:
00009376 BF[47920100]           <1>       mov   edi, Dir_Entry_Name ; Dir Entry Format File Name
0000937B E8D7200000             <1>       call  convert_file_name
00009380 89FE                   <1>       mov   esi, edi ; offset Dir_Entry_Name
                                <1>
00009382 66A1[A6920100]         <1>       mov   ax, [FindFile_AttributesMask]
                                <1>       ;xor  ecx, ecx
00009388 30C9                   <1>       xor   cl, cl
0000938A E8D01D0000             <1>       call  locate_current_dir_file
0000938F 726E                   <1>       jc    short loc_fff_retn
                                <1>       ; EDI = Directory Entry
                                <1>       ; EBX = Directory Buffer Entry Index/Number
                                <1>
                                <1> loc_fff_fnf_ln_check:
00009391 30ED                   <1>       xor   ch, ch
00009393 80F60F                 <1>       xor   dh, 0Fh
00009396 7408                   <1>       jz    short loc_fff_longname_yes
00009398 882D[A5920100]         <1>       mov   [FindFile_LongNameYes], ch ; 0
0000939E EB0C                   <1>       jmp   short loc_fff_longname_no
                                <1>
                                <1> loc_fff_longname_yes:
                                <1>       ;inc  byte [FindFile_LongNameYes]
000093A0 8A0D[B2910100]         <1>       mov   cl, [LFN_EntryLength]
000093A6 880D[A5920100]         <1>       mov   [FindFile_LongNameEntryLength], cl ; FindFile_LongNameYes
                                <1>
                                <1> loc_fff_longname_no:
                                <1>       ;mov  bx, [DirBuff_CurrentEntry]
000093AC 66891D[D0920100]       <1>       mov   [FindFile_DirEntryNumber], bx
000093B3 6689C2                 <1>       mov   dx, ax ; Ambiguous Filename chars used sign > 0
                                <1>
000093B6 A0[B6890100]           <1>       mov   al, [Current_Drv]
000093BB A2[56920100]           <1>       mov   [FindFile_Drv], al
                                <1>
000093C0 A1[B0890100]           <1>       mov   eax, [Current_Dir_FCluster]
000093C5 A3[C8920100]           <1>       mov   [FindFile_DirFirstCluster], eax
                                <1>
000093CA A1[E1900100]           <1>       mov   eax, [DirBuff_Cluster]
000093CF A3[CC920100]           <1>       mov   [FindFile_DirCluster], eax
                                <1>
000093D4 66FF05[D2920100]       <1>       inc   word [FindFile_MatchCounter]
                                <1>
000093DB 89FB                   <1>       mov   ebx, edi
000093DD 89FE                   <1>       mov   esi, edi
000093DF BF[A8920100]           <1>       mov   edi, FindFile_DirEntry
000093E4 89F8                   <1>       mov   eax, edi
000093E6 B108                   <1>       mov   cl, 8
000093E8 F3A5                   <1>       rep   movsd
000093EA 89C6                   <1>       mov   esi, eax
000093EC 89DF                   <1>       mov   edi, ebx
                                <1>
000093EE A1[C4920100]           <1>       mov   eax, [FindFile_DirEntry+28] ; File Size
                                <1>
000093F3 8A1D[B3920100]         <1>       mov   bl, [FindFile_DirEntry+11] ; File Attributes
000093F9 8A3D[A5920100]         <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:
000093FF 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>
00009400 66833D[D2920100]00     <1>       cmp    word [FindFile_MatchCounter], 0
00009408 7707                   <1>       ja     short loc_start_search_next_file
                                <1>
                                <1> loc_fnf_stc_retn:
0000940A F9                     <1>       stc
                                <1> loc_fnf_ax12h_retn:
0000940B B80C000000             <1>       mov    eax, 12 ; No More files
                                <1> ;loc_fnf_retn:
00009410 C3                     <1>       retn
                                <1>
                                <1> loc_start_search_next_file:
00009411 668B1D[D0920100]       <1>       mov    bx, [FindFile_DirEntryNumber]
00009418 6643                   <1>       inc    bx
0000941A 663B1D[DF900100]       <1>       cmp    bx, [DirBuff_LastEntry]
00009421 7719                   <1>       ja     short loc_cont_search_next_file
                                <1>
                                <1> loc_fnf_search:
00009423 BE[47920100]           <1>       mov    esi, Dir_Entry_Name
00009428 66A1[A6920100]         <1>       mov    ax, [FindFile_AttributesMask]
0000942E 6631C9                 <1>       xor    cx, cx
00009431 E82D1E0000             <1>       call   find_directory_entry
00009436 0F8355FFFFFF           <1>         jnc     loc_fff_fnf_ln_check
                                <1>
                                <1> loc_cont_search_next_file:
0000943C 31DB                   <1>       xor    ebx, ebx
0000943E 8A3D[B6890100]         <1>       mov    bh, [Current_Drv]
00009444 BE00010900             <1>       mov    esi, Logical_DOSDisks
00009449 01DE                   <1>       add    esi, ebx
                                <1>
0000944B 803D[B4890100]00       <1>       cmp    byte [Current_Dir_Level], 0
00009452 7608                   <1>       jna    short loc_fnf_check_FAT_type
00009454 807E0301               <1>       cmp    byte [esi+LD_FATType], 1
00009458 72B1                   <1>       jb     short loc_fnf_ax12h_retn
0000945A EB06                   <1>       jmp    short loc_fnf_check_next_cluster
                                <1>
                                <1> loc_fnf_check_FAT_type:
0000945C 807E0303               <1>       cmp    byte [esi+LD_FATType], 3
00009460 72A9                   <1>       jb     short loc_fnf_ax12h_retn
                                <1>
                                <1> loc_fnf_check_next_cluster:
00009462 A1[E1900100]           <1>       mov    eax, [DirBuff_Cluster]
00009467 E8CA370000             <1>       call   get_next_cluster
0000946C 7306                   <1>       jnc    short loc_fnf_load_next_dir_cluster
0000946E 09C0                   <1>       or     eax, eax
00009470 7498                   <1>       jz     short loc_fnf_stc_retn
                                <1>       ;mov   eax, 17 ;Drive not ready or read error
00009472 F5                     <1>       cmc    ;stc
                                <1> loc_fnf_retn:
00009473 C3                     <1>       retn
                                <1>
                                <1> loc_fnf_load_next_dir_cluster:
00009474 E8A3390000             <1>       call   load_FAT_sub_directory
00009479 72F8                   <1>       jc     short loc_fnf_retn
0000947B 6631DB                 <1>       xor    bx, bx
0000947E 66891D[D0920100]       <1>       mov    [FindFile_DirEntryNumber], bx
00009485 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:
00009487 803E20                 <1>       cmp    byte [esi], 20h
0000948A 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:
0000948C C3                     <1>       retn
                                <1> loc_find_longname:
0000948D E839210000             <1>       call   find_longname
00009492 7328                   <1>       jnc    short loc_print_longname
                                <1>
00009494 08C0                   <1>       or     al, al
00009496 741A                   <1>       jz     short loc_longname_not_found
```

```
                                <1>
                                <1>       ; 16/10/2016 (15h -> 15, 17)
00009498 3C0F                   <1>       cmp    al, 15
0000949A 0F84B6F7FFFF           <1>       je     cd_drive_not_ready ; drive not ready
                                <1>                                 ; or
000094A0 3C11                   <1>       cmp    al, 17          ; read error
000094A2 0F84AEF7FFFF           <1>       je     cd_drive_not_ready
                                <1>
                                <1> loc_ln_file_dir_not_found:
000094A8 BE[AE420100]           <1>       mov    esi, Msg_File_Directory_Not_Found
                                <1>       ;call  print_msg
                                <1>         ;retn
000094AD E973DFFFFF             <1>       jmp    print_msg
                                <1>
                                <1> loc_longname_not_found:
000094B2 BE[CD420100]           <1>         mov     esi, Msg_LongName_Not_Found
                                <1>       ;call  print_msg
                                <1>         ;retn
000094B7 E969DFFFFF             <1>       jmp    print_msg
                                <1>
                                <1> loc_print_longname:
                                <1>       ;mov   esi, LongFileName
000094BC BF[B68A0100]           <1>       mov    edi, TextBuffer
000094C1 57                     <1>       push   edi
000094C2 3C00                   <1>       cmp    al, 0
000094C4 7708                   <1>       ja     short loc_print_longname_1
                                <1> loc_print_FS_longname: ; Singlix FS (64 byte ASCIIZ file name)
000094C6 AC                     <1>       lodsb
000094C7 AA                     <1>       stosb
000094C8 08C0                   <1>       or     al, al
000094CA 75FA                   <1>       jnz    short loc_print_FS_longname
000094CC EB07                   <1>       jmp    short loc_print_longname_2
                                <1>       ;
                                <1> loc_print_longname_1: ; MS Windows long name (UNICODE chars)
000094CE 66AD                   <1>       lodsw
000094D0 AA                     <1>       stosb
000094D1 08C0                   <1>       or     al, al
000094D3 75F9                   <1>       jnz    short loc_print_longname_1
                                <1>       ;
                                <1> loc_print_longname_2:
000094D5 5E                     <1>       pop    esi
000094D6 E84ADFFFFF             <1>       call   print_msg
000094DB BE[1F4C0100]           <1>       mov    esi, nextline
                                <1>       ;call  print_msg
                                <1>       ;retn
000094E0 E940DFFFFF             <1>       jmp    print_msg
                                <1>
                                <1> show_file:
                                <1>       ; 18/02/2016
                                <1>       ; 17/02/2016
                                <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 13/09/2011 (CMD_INTR.ASM, 'cmp_cmd_show')
                                <1>       ; 08/11/2009
                                <1>
                                <1> loc_show_parse_path_name:
000094E5 BF[56920100]           <1>       mov    edi, FindFile_Drv
000094EA E833200000             <1>       call   parse_path_name
000094EF 0F824CF9FFFF           <1>       jc     loc_cmd_failed
                                <1>
                                <1> loc_show_check_filename_exists:
000094F5 BE[98920100]           <1>       mov    esi, FindFile_Name
000094FA 803E20                 <1>       cmp    byte [esi], 20h
000094FD 0F863EF9FFFF           <1>       jna    loc_cmd_failed
                                <1>
                                <1>       ; 15/02/2016 (invalid file name check)
00009503 E807020000             <1>       call   check_filename
00009508 730A                   <1>       jnc    short loc_show_change_drv
                                <1>
0000950A BE[9A430100]           <1>       mov    esi, Msg_invalid_name_chars
0000950F E911DFFFFF             <1>       jmp    print_msg
                                <1>
                                <1> loc_show_change_drv:
00009514 8A35[B6890100]         <1>       mov    dh, [Current_Drv]
0000951A 8835[12910100]         <1>       mov    [RUN_CDRV], dh
00009520 8A15[56920100]         <1>       mov    dl, [FindFile_Drv]
00009526 38F2                   <1>       cmp    dl, dh
00009528 740B                   <1>       je     short loc_show_change_directory
0000952A E87FEAFFFF             <1>       call   change_current_drive
                                <1>       ;jc    loc_file_rw_cmd_failed
0000952F 0F8237F9FFFF           <1>       jc     loc_run_cmd_failed
                                <1>
                                <1> loc_show_change_directory:
00009535 803D[57920100]20       <1>       cmp    byte [FindFile_Directory], 20h
0000953C 7618                   <1>       jna    short loc_findload_showfile
                                <1>
0000953E FE05[833F0100]         <1>       inc    byte [Restore_CDIR]
00009544 BE[57920100]           <1>       mov    esi, FindFile_Directory
00009549 30E4                   <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000954B E8BC190000             <1>       call   change_current_directory
                                <1>       ;jc    loc_file_rw_cmd_failed
00009550 0F8216F9FFFF           <1>       jc     loc_run_cmd_failed
                                <1>
                                <1> ;loc_show_change_prompt_dir_string:
                                <1>       ;call  change_prompt_dir_string
                                <1>
                                <1> loc_findload_showfile:
                                <1>       ; 15/02/2016
00009556 BE[98920100]           <1>       mov    esi, FindFile_Name
0000955B BF[47920100]           <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
00009560 E8F21E0000             <1>       call   convert_file_name
00009565 89FE                   <1>       mov    esi, edi ; offset Dir_Entry_Name
                                <1>
00009567 28C0                   <1>       sub    al, al ; Attrib AND mask = 0
                                <1>       ; Directory attribute : 10h
                                <1>       ; Volume name attribute: 8h
```

```
5767 00009569 B418              <1>       mov    ah, 00011000b ; 18h (Attrib NAND, AND --> zero mask)
5768                            <1>       ;
5769 0000956B 6631C9            <1>       xor    cx, cx
5770 0000956E E8EC1B0000        <1>       call   locate_current_dir_file
5771                            <1>       ;jc    loc_file_rw_cmd_failed
5772 00009573 0F82F3F8FFFF      <1>       jc     loc_run_cmd_failed
5773                            <1>
5774                            <1> loc_show_load_file:
5775                            <1>       ; EDI = Directory Entry
5776 00009579 668B4714          <1>       mov    ax, [edi+DirEntry_FstClusHI] ; First Cluster High Word
5777 0000957D C1E010            <1>       shl    eax, 16
5778 00009580 668B471A          <1>       mov    ax, [edi+DirEntry_FstClusLO] ; First Cluster Low Word
5779 00009584 A3[00930100]      <1>       mov    [Show_Cluster], eax
5780 00009589 8B471C            <1>       mov    eax, [edi+DirEntry_FileSize] ; File Size
5781 0000958C 21C0              <1>       and    eax, eax ; Empty file !
5782 0000958E 0F8491000000      <1>         jz     end_of_show_file
5783 00009594 A3[04930100]      <1>       mov    [Show_FileSize], eax
5784 00009599 31C0              <1>       xor    eax, eax
5785 0000959B A3[08930100]      <1>       mov    [Show_FilePointer], eax ; 0
5786 000095A0 66A3[0C930100]    <1>       mov    [Show_ClusterPointer], ax ; 0
5787 000095A6 29DB              <1>       sub    ebx, ebx
5788 000095A8 8A3D[B6890100]    <1>       mov    bh, [Current_Drv]
5789 000095AE BE00010900        <1>       mov    esi, Logical_DOSDisks
5790 000095B3 01DE              <1>       add    esi, ebx
5791 000095B5 8935[FC920100]    <1>       mov    [Show_LDDDT], esi ; Logical DOS Drv Description Table addr
5792                            <1>
5793 000095BB 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
5794 000095BF 7713              <1>       ja     short loc_show_calculate_cluster_size
5795                            <1>       ; Singlix FS
5796                            <1>       ; First Cluster Number is FDT number (in compatibility buffer)
5797 000095C1 8B15[00930100]    <1>       mov    edx, [Show_Cluster] ; Compatibility dir. buffer value (FDT)
5798 000095C7 8915[F8920100]    <1>       mov    [Show_FDT], edx
5799 000095CD 31C0              <1>       xor    eax, eax
5800 000095CF A3[00930100]      <1>       mov    [Show_Cluster], eax ; Sector index  = 0
5801                            <1>                                  ; (next time it will be 1)
5802                            <1> loc_show_calculate_cluster_size:
5803 000095D4 668B5E11          <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec] ; FAT 12-16-32 (512)
5804                            <1>       ; BX = 512 = [esi+LD_FS_BytesPerSec] ; Singlix FS
5805 000095D8 8A4613            <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust] ; FAT 12-16-32 (<= 128)
5806                            <1>       ; AL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
5807 000095DB F7E3              <1>       mul    ebx
5808                            <1>
5809                            <1>       ;cmp   eax, 65536 ; non-compatible (very big) cluster size
5810                            <1>       ;ja    short end_of_show_file
5811 000095DD 66A3[0E930100]    <1>       mov    [Show_ClusterSize], ax
5812                            <1>
5813                            <1> loc_start_show_file:
5814 000095E3 BE[1F4C0100]      <1>       mov    esi, nextline
5815 000095E8 E838DEFFFF        <1>       call   print_msg
5816                            <1>
5817 000095ED A1[00930100]      <1>       mov    eax, [Show_Cluster]
5818 000095F2 C605[10930100]17  <1>       mov    byte [Show_RowCount], 23
5819                            <1>
5820                            <1>       ; 17/02/2016
5821 000095F9 8B35[FC920100]    <1>       mov    esi, [Show_LDDDT]
5822                            <1>
5823                            <1> loc_show_next_cluster:
5824                            <1>       ; 15/02/2016
5825 000095FF BB00000700        <1>       mov    ebx, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
5826                            <1>       ; ESI = Logical DOS drv description table address
5827 00009604 E851380000        <1>       call   read_cluster
5828                            <1>       ;jc    loc_file_rw_cmd_failed
5829 00009609 0F825DF8FFFF      <1>       jc     loc_run_cmd_failed
5830                            <1>
5831 0000960F 31DB              <1>       xor    ebx, ebx
5832                            <1> loc_show_next_byte:
5833 00009611 803D[10930100]00  <1>       cmp    byte [Show_RowCount], 0
5834 00009618 7521              <1>       jne    short pass_show_wait_for_key
5835 0000961A 30E4              <1>       xor    ah, ah
5836 0000961C E8C278FFFF        <1>       call   int16h
5837 00009621 3C1B              <1>       cmp    al, 1Bh
5838 00009623 750F              <1>       jne    short pass_exit_show
5839                            <1> end_of_show_file:
5840                            <1> pass_show_file:
5841 00009625 BE[1F4C0100]      <1>       mov    esi, nextline
5842 0000962A E8F6DDFFFF        <1>       call   print_msg
5843 0000962F E94B010000        <1>       jmp    loc_file_rw_restore_retn
5844                            <1>
5845                            <1> pass_exit_show:
5846 00009634 C605[10930100]14  <1>       mov    byte [Show_RowCount], 20
5847                            <1> pass_show_wait_for_key:
5848 0000963B 81C300000700      <1>       add    ebx, Cluster_Buffer
5849 00009641 8A03              <1>       mov    al, [ebx]
5850 00009643 3C0D              <1>       cmp    al, 0Dh
5851 00009645 0F8590000000      <1>         jne    loc_show_check_tab_space
5852 0000964B FE0D[10930100]    <1>       dec    byte [Show_RowCount]
5853                            <1> pass_show_dec_rowcount:
5854 00009651 B307              <1>       mov    bl, 7 ; (light gray character color, black background)
5855 00009653 8A3D[1E890100]    <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
5856 00009659 E8248CFFFF        <1>       call   _write_tty
5857                            <1> loc_show_check_eof:
5858 0000965E FF05[08930100]    <1>       inc    dword [Show_FilePointer]
5859 00009664 A1[08930100]      <1>       mov    eax, [Show_FilePointer]
5860 00009669 3B05[04930100]    <1>       cmp    eax, [Show_FileSize]
5861 0000966F 73B4              <1>       jnb    short end_of_show_file
5862 00009671 66FF05[0C930100]  <1>       inc    word [Show_ClusterPointer]
5863 00009678 0FB71D[0C930100]  <1>       movzx  ebx, word [Show_ClusterPointer]
5864                            <1>
5865                            <1>       ; 17/02/2016
5866                            <1>       ; (sector boundary -9 bits- check, 512 = 0)
5867 0000967F 66F7C3FF01        <1>         test    bx, 1FFh ;  1 to 511
5868 00009684 758B              <1>       jnz    short loc_show_next_byte
5869                            <1>
5870                            <1>       ; 16/02/2016
5871 00009686 8B35[FC920100]    <1>       mov    esi, [Show_LDDDT]
```

```
5872                                    <1>       ;
5873 0000968C 807E0300                  <1>       cmp    byte [esi+LD_FATType], 0
5874 00009690 7719                      <1>       ja     short loc_show_check_fat_cluster_size
5875                                    <1>
5876                                    <1>       ; Singlix FS
5877                                    <1>       ; 1 sector, more... (cluster size = 1 sector)
5878 00009692 A1[00930100]              <1>       mov    eax, [Show_Cluster]
5879 00009697 40                        <1>       inc    eax
5880 00009698 A3[00930100]              <1>       mov    [Show_Cluster], eax
5881                                    <1>
5882 0000969D 6621DB                    <1>       and    bx, bx ; 65536 -> 0
5883 000096A0 0F856BFFFFFF              <1>         jnz  loc_show_next_byte
5884 000096A6 E954FFFFFF                <1>       jmp     loc_show_next_cluster
5885                                    <1>
5886                                    <1> loc_show_check_fat_cluster_size:
5887                                    <1>       ; 17/02/2016
5888 000096AB 663B1D[0E930100]          <1>       cmp    bx, [Show_ClusterSize] ; cluster size in bytes
5889 000096B2 0F8259FFFFFF              <1>         jb   loc_show_next_byte
5890 000096B8 66C705[0C930100]00-       <1>       mov    word [Show_ClusterPointer], 0
5890 000096C0 00                        <1>
5891                                    <1>
5892 000096C1 A1[00930100]              <1>       mov    eax, [Show_Cluster]
5893                                    <1>       ;mov   esi, [Show_LDDDT]
5894                                    <1> loc_show_get_next_cluster:
5895 000096C6 E86B350000                <1>       call   get_next_cluster
5896                                    <1>       ;jc    loc_file_rw_cmd_failed
5897 000096CB 0F829BF7FFFF              <1>       jc     loc_run_cmd_failed
5898                                    <1> loc_show_update_ccluster:
5899 000096D1 A3[00930100]              <1>       mov    [Show_Cluster], eax
5900 000096D6 E924FFFFFF                <1>         jmp     loc_show_next_cluster
5901                                    <1>
5902                                    <1> loc_show_check_tab_space:
5903 000096DB 3C09                      <1>       cmp    al, 09h
5904 000096DD 0F856EFFFFFF              <1>         jne     pass_show_dec_rowcount
5905                                    <1> loc_show_put_tab_space:
5906 000096E3 8A3D[1E890100]            <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
5907 000096E9 E82188FFFF                <1>       call   get_cpos
5908                                    <1>       ; dl = cursor column
5909 000096EE 80E207                    <1>       and    dl, 7 ; 18/02/2016
5910                                    <1>       ;shr   bh, 1 ; [ACTIVE_PAGE]
5911 000096F1 8A3D[1E890100]            <1>       mov    bh, [ACTIVE_PAGE]
5912 000096F7 B307                      <1>       mov    bl, 7 ; color attribute
5913                                    <1> loc_show_put_space_chars:
5914 000096F9 B020                      <1>       mov    al, 20h ; space
5915                                    <1>       ;mov   bh, [ACTIVE_PAGE] ; [ptty]
5916                                    <1>       ;mov   bl, 7 ; color attribute
5917                                    <1>       ;push  dx
5918 000096FB 52                        <1>       push   edx ; 29/12/2017
5919 000096FC E8818BFFFF                <1>       call   _write_tty
5920 00009701 5A                        <1>       pop    edx ; 29/12/2017
5921                                    <1>       ;pop   dx
5922                                    <1>       ; 18/02/2016
5923 00009702 80FA07                    <1>       cmp    dl, 7
5924 00009705 0F8353FFFFFF              <1>       jnb    loc_show_check_eof
5925 0000970B FEC2                      <1>       inc    dl
5926 0000970D EBEA                      <1>       jmp    short loc_show_put_space_chars
5927                                    <1>
5928                                    <1> check_filename:
5929                                    <1>       ; 10/10/2016
5930                                    <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
5931                                    <1>       ; 07/08/2010 (FILE.ASM, 'proc_check_filename')
5932                                    <1>       ; 10/07/2010
5933                                    <1>       ; Derived from 'proc_check_filename'
5934                                    <1>       ; in the old TRDOS.ASM (09/02/2005).
5935                                    <1>       ;
5936                                    <1>       ; INPUT ->
5937                                    <1>       ;      ESI = Dot File Name Location
5938                                    <1>       ; OUTPUT ->
5939                                    <1>       ;      cf = 1 -> error code in AL
5940                                    <1>       ;           AL = ERR_INV_FILE_NAME (=26)
5941                                    <1>       ;               Invalid file name chars
5942                                    <1>       ;      cf = 0 -> valid file name
5943                                    <1>       ;
5944                                    <1>       ;(EAX, ECX, EDI will be changed)
5945                                    <1>
5946                                    <1> check_invalid_filename_chars:
5947                                    <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
5948                                    <1>       ; 10/07/2010 (FILE.ASM, 'proc_check_invalid_filename_chars')
5949                                    <1>       ; 10/02/2010
5950                                    <1>       ; Derived from 'proc_check_invalid_filename_chars'
5951                                    <1>       ; in the old TRDOS.ASM (09/02/2005).
5952                                    <1>       ;
5953                                    <1>       ; INPUT ->
5954                                    <1>       ;      ESI = ASCIIZ FileName
5955                                    <1>       ; OUTPUT ->
5956                                    <1>       ;      cf = 1 -> invalid
5957                                    <1>       ;      cf = 0 -> valid
5958                                    <1>       ;
5959                                    <1>       ;(EAX, ECX, EDI will be changed)
5960                                    <1>
5961 0000970F 56                        <1>       push   esi
5962                                    <1>
5963 00009710 BF[82400100]              <1>         mov     edi, invalid_fname_chars
5964 00009715 AC                        <1>       lodsb
5965                                    <1> check_filename_next_char:
5966 00009716 B914000000                <1>       mov    ecx, sizeInvFnChars
5967 0000971B BF[82400100]              <1>       mov    edi, invalid_fname_chars
5968                                    <1> loc_scan_invalid_filename_char:
5969 00009720 AE                        <1>       scasb
5970 00009721 741F                      <1>       je     short loc_invalid_filename_stc
5971 00009723 E2FB                      <1>       loop   loc_scan_invalid_filename_char
5972 00009725 AC                        <1>       lodsb
5973 00009726 3C1F                      <1>       cmp    al, 1Fh  ; 20h and above
5974 00009728 77EC                      <1>       ja     short check_filename_next_char
5975                                    <1>
```

```
                                <1> check_filename_dot:
0000972A 8B3424                 <1>         mov    esi, [esp]
                                <1>
0000972D B421                   <1>         mov    ah, 21h
0000972F B908000000             <1>         mov    ecx, 8
                                <1> loc_check_filename_next_char:
00009734 AC                     <1>         lodsb
00009735 3C2E                   <1>         cmp    al, 2Eh
00009737 7511                   <1>         jne    short pass_check_fn_dot_check
                                <1> loc_check_filename_ext_0:
00009739 AC                     <1>         lodsb
0000973A 38E0                   <1>         cmp    al, ah ; 21h
0000973C 7205                   <1>         jb     short loc_invalid_filename
0000973E 3C2E                   <1>         cmp    al, 2Eh
00009740 7519                   <1>         jne    short loc_check_filename_ext_1
                                <1>
                                <1> loc_invalid_filename_stc:
                                <1> loc_check_fn_stc_rtn:
00009742 F9                     <1>         stc
                                <1> loc_invalid_filename:
                                <1>         ; 10/10/2016 (0Bh -> 26)
00009743 B81A000000             <1>         mov    eax, ERR_INV_FILE_NAME ; (=26)
                                <1>         ; Invalid file name chars
                                <1> loc_check_fn_rtn:
00009748 5E                     <1>         pop    esi
00009749 C3                     <1>         retn
                                <1>
                                <1> pass_check_fn_dot_check:
0000974A 38E0                   <1>         cmp    al, ah ; 21h
0000974C 7224                   <1>         jb     short loc_check_fn_clc_rtn
0000974E E2E4                   <1>         loop   loc_check_filename_next_char
00009750 AC                     <1>         lodsb
00009751 38E0                   <1>         cmp    al, ah ; 21h
00009753 721D                   <1>         jb     short loc_check_fn_clc_rtn
00009755 3C2E                   <1>         cmp    al, 2Eh
00009757 75E9                   <1>         jne    short loc_check_fn_stc_rtn
00009759 EBDE                   <1>         jmp    short loc_check_filename_ext_0
                                <1>
                                <1> loc_check_filename_ext_1:
0000975B AC                     <1>         lodsb
0000975C 38E0                   <1>         cmp    al, ah ; 21h
0000975E 7212                   <1>         jb     short loc_check_fn_clc_rtn
00009760 3C2E                   <1>         cmp    al, 2Eh
00009762 74DE                   <1>         je     short loc_check_fn_stc_rtn
00009764 AC                     <1>         lodsb
00009765 38E0                   <1>         cmp    al, ah ; 21h
00009767 7209                   <1>         jb     short loc_check_fn_clc_rtn
00009769 3C2E                   <1>         cmp    al, 2Eh
0000976B 74D5                   <1>         je     short loc_check_fn_stc_rtn
0000976D AC                     <1>         lodsb
0000976E 38E0                   <1>         cmp    al, ah ; 21h
00009770 73D0                   <1>         jnb    short loc_check_fn_stc_rtn
                                <1>
                                <1> loc_check_fn_clc_rtn:
00009772 5E                     <1>         pop    esi
00009773 F8                     <1>         clc
00009774 C3                     <1>         retn
                                <1>
                                <1> loc_print_deleted_message:
00009775 BE[6F440100]           <1>         mov    esi, Msg_Deleted
0000977A E8A6DCFFFF             <1>         call   print_msg
                                <1>
                                <1>         ;clc
                                <1>
                                <1> loc_file_rw_restore_retn:
                                <1>         ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 28/02/2010 (CMD_INTR.ASM)
                                <1> loc_file_rw_cmd_failed:
0000977F 9C                     <1>         pushf
00009780 E84FF7FFFF             <1>         call   restore_cdir_after_cmd_fail
00009785 9D                     <1>         popf
00009786 720D                   <1>         jc     short loc_file_rw_check_write_fault
00009788 C3                     <1>         retn
                                <1>
                                <1> loc_permission_denied:
                                <1>         ; 27/02/2016
00009789 BE[7C440100]           <1>         mov    esi, Msg_Permission_Denied
0000978E E892DCFFFF             <1>         call   print_msg
00009793 EBEA                   <1>         jmp    short loc_file_rw_restore_retn
                                <1>
                                <1> loc_file_rw_check_write_fault:
                                <1>         ;cmp   al, 1Dh ; Write Fault
00009795 3C12                   <1>           cmp  al, 18 ; 05/11/2016
00009797 0F85D4F6FFFF           <1>         jne     loc_run_cmd_failed_cmp_al
0000979D BE[63420100]           <1>         mov    esi, Msg_Not_Ready_Write_Err
                                <1>         ;call  print_msg
                                <1>         ;retn
000097A2 E97EDCFFFF             <1>         jmp    print_msg
                                <1>
                                <1> make_directory:
                                <1>         ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 12/03/2011 (CMD_INTR.ASM, 'cmp_cmd_mkdir')
                                <1>         ; 14/08/2010
                                <1>         ; 10/07/2010
                                <1>         ; 29/11/2009
                                <1>         ;
                                <1> get_mkdir_fchar:
                                <1>         ; esi = directory name
000097A7 803E20                 <1>         cmp    byte [esi], 20h
000097AA 7701                   <1>           ja   short loc_mkdir_parse_path_name
                                <1>
                                <1> loc_mkdir_nodirname_retn:
000097AC C3                     <1>         retn
                                <1>
                                <1> loc_mkdir_parse_path_name:
```

```
6081 000097AD BF[56920100]        <1>       mov    edi, FindFile_Drv
6082 000097B2 E86B1D0000          <1>         call    parse_path_name
6083 000097B7 0F8284F6FFFF        <1>       jc     loc_cmd_failed
6084                              <1>
6085                              <1> loc_mkdir_check_dirname_exists:
6086 000097BD BE[98920100]        <1>       mov    esi, FindFile_Name
6087 000097C2 803E20              <1>       cmp    byte [esi], 20h
6088 000097C5 0F8676F6FFFF        <1>       jna    loc_cmd_failed
6089 000097CB 8935[14930100]      <1>       mov    [DelFile_FNPointer], esi
6090 000097D1 E839FFFFFF          <1>       call   check_filename
6091 000097D6 7259                <1>       jc     short loc_mkdir_invalid_dir_name_chars
6092                              <1>
6093                              <1> loc_mkdir_drv:
6094 000097D8 8A35[B6890100]      <1>       mov    dh, [Current_Drv]
6095 000097DE 8835[12910100]      <1>       mov    [RUN_CDRV], dh
6096                              <1>
6097 000097E4 8A15[56920100]      <1>       mov    dl, [FindFile_Drv]
6098 000097EA 38F2                <1>       cmp    dl, dh
6099 000097EC 7407                <1>       je     short loc_mkdir_change_directory
6100                              <1>
6101 000097EE E8BBE7FFFF          <1>       call   change_current_drive
6102 000097F3 728A                <1>       jc     loc_file_rw_cmd_failed
6103                              <1>
6104                              <1> loc_mkdir_change_directory:
6105 000097F5 803D[57920100]20    <1>       cmp    byte [FindFile_Directory], 20h
6106 000097FC 7614                <1>       jna    short loc_mkdir_find_directory
6107                              <1>
6108 000097FE FE05[833F0100]      <1>       inc    byte [Restore_CDIR]
6109 00009804 BE[57920100]        <1>       mov    esi, FindFile_Directory
6110 00009809 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
6111 0000980B E8FC160000          <1>       call   change_current_directory
6112 00009810 722E                <1>       jc     short loc_mkdir_check_error_code
6113                              <1>
6114                              <1> ;loc_mkdir_change_prompt_dir_string:
6115                              <1>       ;call  change_prompt_dir_string
6116                              <1>
6117                              <1> loc_mkdir_find_directory:
6118                              <1>       ;mov   esi, FindFile_Name
6119 00009812 8B35[14930100]      <1>       mov    esi, [DelFile_FNPointer]
6120                              <1>       ;xor   eax, eax
6121 00009818 6631C0              <1>       xor    ax, ax ; any name (dir, file, volume)
6122 0000981B E831FBFFFF          <1>       call   find_first_file
6123 00009820 721E                <1>       jc     short loc_mkdir_check_error_code
6124                              <1>
6125                              <1> loc_mkdir_directory_found:
6126 00009822 BE[C7430100]        <1>       mov    esi, Msg_Name_Exists
6127 00009827 E8F9DBFFFF          <1>       call   print_msg
6128                              <1>
6129 0000982C E94EFFFFFF          <1>         jmp     loc_file_rw_restore_retn
6130                              <1>
6131                              <1> loc_mkdir_invalid_dir_name_chars:
6132 00009831 BE[9A430100]        <1>       mov    esi, Msg_invalid_name_chars
6133 00009836 E8EADBFFFF          <1>       call   print_msg
6134                              <1>
6135 0000983B E93FFFFFFF          <1>         jmp     loc_file_rw_restore_retn
6136                              <1>
6137                              <1> loc_mkdir_check_error_code:
6138 00009840 3C02                <1>       cmp    al, 2
6139                              <1>       ;je    short loc_mkdir_directory_not_found
6140 00009842 7406                <1>       je     short loc_mkdir_ask_for_yes_no
6141 00009844 F9                  <1>       stc
6142 00009845 E935FFFFFF          <1>         jmp     loc_file_rw_cmd_failed
6143                              <1>
6144                              <1> loc_mkdir_directory_not_found:
6145                              <1> loc_mkdir_ask_for_yes_no:
6146 0000984A BE[E8430100]        <1>       mov    esi, Msg_DoYouWantMkdir
6147 0000984F E8D1DBFFFF          <1>       call   print_msg
6148 00009854 8B35[14930100]      <1>       mov    esi, [DelFile_FNPointer]
6149 0000985A E8C6DBFFFF          <1>       call   print_msg
6150 0000985F BE[07440100]        <1>       mov    esi, Msg_YesNo
6151 00009864 E8BCDBFFFF          <1>       call   print_msg
6152                              <1>
6153 00009869 C605[11440100]20    <1>       mov    byte [Y_N_nextline], 20h
6154                              <1>
6155                              <1> loc_mkdir_ask_again:
6156 00009870 30E4                <1>       xor    ah, ah
6157 00009872 E86C76FFFF          <1>       call   int16h
6158 00009877 3C1B                <1>       cmp    al, 1Bh
6159                              <1>       ;je    short loc_do_not_make_directory
6160 00009879 7439                <1>       je     short loc_mkdir_y_n_escape
6161 0000987B 24DF                <1>       and    al, 0DFh ; y -> Y, n -> N
6162 0000987D 3C59                <1>       cmp    al, 'Y' ; 'yes'
6163 0000987F 7404                <1>       je     short loc_mkdir_yes_make_directory
6164 00009881 3C4E                <1>       cmp    al, 'N' ; 'no'
6165 00009883 75EB                <1>       jne    short loc_mkdir_ask_again
6166                              <1>
6167                              <1> loc_do_not_make_directory:
6168                              <1> loc_mkdir_yes_make_directory:
6169 00009885 E82E000000          <1>       call   y_n_answer ; 29/12/2017
6170                              <1>       ;cmp   al, 'Y' ; 'yes'
6171                              <1>       ;cmc
6172                              <1>         ;jnc loc_file_rw_restore_retn
6173 0000988A 3C4E                <1>       cmp    al, 'N' ; 'no'
6174 0000988C 0F84EDFEFFFF        <1>         je   loc_file_rw_restore_retn
6175                              <1>
6176                              <1> loc_mkdir_call_make_sub_directory:
6177 00009892 8B35[14930100]      <1>       mov    esi, [DelFile_FNPointer]
6178 00009898 B110                <1>       mov    cl, 10h ; Directory attributes
6179 0000989A E8821D0000          <1>       call   make_sub_directory
6180                              <1> loc_rename_file_ok: ; 06/03/2016
6181 0000989F 0F82DAFEFFFF        <1>         jc   loc_file_rw_cmd_failed
6182                              <1> move_source_file_to_destination_OK:
6183 000098A5 BE[15440100]        <1>       mov    esi, Msg_OK
6184 000098AA E876DBFFFF          <1>       call   print_msg
6185 000098AF E9CBFEFFFF          <1>       jmp    loc_file_rw_restore_retn
```

```
                                <1>
                                <1> loc_mkdir_y_n_escape:
000098B4 B04E                   <1>       mov    al, 'N' ; 'no'
000098B6 EBCD                   <1>       jmp    short loc_do_not_make_directory
                                <1>
                                <1> y_n_answer:
                                <1>       ; 29/12/2017
000098B8 A2[11440100]           <1>       mov    [Y_N_nextline], al
                                <1>       ;push  ax
000098BD 50                     <1>       push   eax
000098BE BE[11440100]           <1>       mov    esi, Y_N_nextline
000098C3 E85DDBFFFF             <1>       call   print_msg
000098C8 58                     <1>       pop    eax
                                <1>       ;pop   ax
000098C9 C3                     <1>       retn
                                <1>
                                <1> delete_directory:
                                <1>       ; 29/12/2017
                                <1>       ; 15/10/2016
                                <1>       ; 01/03/2016, 06/03/2016
                                <1>       ; 27/02/2016, 28/02/2016, 29/02/2016
                                <1>       ; 26/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 16/10/2010 (CMD_INTR.ASM, 'cmp_cmd_rmdir')
                                <1>       ; 05/06/2010
                                <1>       ;
                                <1> get_fchar:
                                <1>       ; esi = directory name
000098CA 803E20                 <1>       cmp    byte [esi], 20h
000098CD 7701                   <1>         ja   short loc_rmdir_parse_path_name
                                <1>
                                <1> loc_rmdir_nodirname_retn:
000098CF C3                     <1>       retn
                                <1>
                                <1> loc_rmdir_parse_path_name:
000098D0 BF[56920100]           <1>       mov    edi, FindFile_Drv
000098D5 E8481C0000             <1>       call   parse_path_name
000098DA 0F8261F5FFFF           <1>       jc     loc_cmd_failed
                                <1>
                                <1> loc_rmdir_check_dirname_exists:
000098E0 BE[98920100]           <1>       mov    esi, FindFile_Name
000098E5 803E20                 <1>       cmp    byte [esi], 20h
000098E8 0F8653F5FFFF           <1>       jna    loc_cmd_failed
000098EE 8935[14930100]         <1>       mov    [DelFile_FNPointer], esi
                                <1>
                                <1> loc_rmdir_drv:
000098F4 8A35[B6890100]         <1>       mov    dh, [Current_Drv]
000098FA 8835[12910100]         <1>       mov    [RUN_CDRV], dh
                                <1>
00009900 8A15[56920100]         <1>       mov    dl, [FindFile_Drv]
00009906 38F2                   <1>       cmp    dl, dh
00009908 740B                   <1>       je     short loc_rmdir_change_directory
                                <1>
0000990A E89FE6FFFF             <1>       call   change_current_drive
0000990F 0F826AFEFFFF           <1>       jc     loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rmdir_change_directory:
00009915 803D[57920100]20       <1>       cmp    byte [FindFile_Directory], 20h
0000991C 7614                   <1>       jna    short loc_rmdir_find_directory
                                <1>
0000991E FE05[833F0100]         <1>       inc    byte [Restore_CDIR]
00009924 BE[57920100]           <1>       mov    esi, FindFile_Directory
00009929 30E4                   <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000992B E8DC150000             <1>       call   change_current_directory
00009930 7211                   <1>       jc     short loc_rmdir_check_error_code
                                <1>
                                <1> ;loc_rmdir_change_prompt_dir_string:
                                <1>       ;call  change_prompt_dir_string
                                <1>
                                <1> loc_rmdir_find_directory:
                                <1>       ;mov   esi, FindFile_Name
00009932 8B35[14930100]         <1>       mov    esi, [DelFile_FNPointer]
00009938 66B81008               <1>       mov    ax, 0810h ; Only directories
0000993C E810FAFFFF             <1>       call   find_first_file
00009941 730A                   <1>       jnc    short loc_rmdir_ambgfn_check
                                <1>
                                <1> loc_rmdir_check_error_code:
00009943 3C02                   <1>       cmp    al, 2
00009945 740B                   <1>       je     short loc_rmdir_directory_not_found
00009947 F9                     <1>       stc
00009948 E932FEFFFF             <1>       jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rmdir_ambgfn_check:
0000994D 6621D2                 <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
00009950 740F                   <1>       jz     short loc_rmdir_directory_found
                                <1>
                                <1> loc_rmdir_directory_not_found:
00009952 BE[85420100]           <1>       mov    esi, Msg_Dir_Not_Found
00009957 E8C9DAFFFF             <1>       call   print_msg
                                <1>
0000995C E91EFEFFFF             <1>       jmp    loc_file_rw_restore_retn
                                <1>
                                <1> loc_rmdir_directory_found:
00009961 80E307                 <1>       and    bl, 07h ; Attributes
00009964 0F851FFEFFFF           <1>       jnz    loc_permission_denied
                                <1>
                                <1> loc_rmdir_save_lnel: ; 28/02/2016
                                <1>        ;mov  bh, [LongName_EntryLength]
0000996A 883D[1E930100]         <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
                                <1>       ; edi = Directory Entry Offset (DirBuff)
                                <1>       ; esi = Directory Entry (FFF Structure)
                                <1>       ;mov   [DelFile_DirEntryAddr], edi ; not required
                                <1>       ;mov   ax, [edi+20] ; First Cluster High Word
                                <1>         ;shl eax, 16
                                <1>       ;mov   ax, [edi+26] ; First Cluster Low Word
                                <1>       ; ROOT Dir First Cluster = 0
```

```
                                <1>         ;cmp eax, 2
                                <1>       ;jb   loc_update_direntry_1
                                <1>
                                <1> pass_rmdir_fc_check:
00009970 57                     <1>       push  edi ; * (29/02/2016)
                                <1>
00009971 BE[1B440100]           <1>       mov   esi, Msg_DoYouWantRmDir
00009976 E8AADAFFFF             <1>       call  print_msg
0000997B 8B35[14930100]         <1>       mov   esi, [DelFile_FNPointer]
00009981 E89FDAFFFF             <1>       call  print_msg
00009986 BE[07440100]           <1>       mov   esi, Msg_YesNo
0000998B E895DAFFFF             <1>       call  print_msg
                                <1>
                                <1> loc_rmdir_ask_again:
00009990 30E4                   <1>       xor   ah, ah
00009992 E84C75FFFF             <1>       call  int16h
00009997 3C1B                   <1>       cmp   al, 1Bh
                                <1>       ;je   short loc_do_not_delete_directory
00009999 7433                   <1>         je      loc_rmdir_y_n_escape ; 06/03/2016
0000999B 24DF                   <1>       and   al, 0DFh
0000999D A2[11440100]           <1>       mov   [Y_N_nextline], al
000099A2 3C59                   <1>       cmp   al, 'Y'
000099A4 7404                   <1>       je    short loc_rmdir_yes_delete_directory
000099A6 3C4E                   <1>       cmp   al, 'N'
000099A8 75E6                   <1>       jne   short loc_rmdir_ask_again
                                <1>
                                <1> loc_do_not_delete_directory:
                                <1> loc_rmdir_yes_delete_directory:
000099AA E809FFFFFF             <1>       call  y_n_answer ; 29/12/2017
000099AF 5F                     <1>       pop   edi ; * (29/02/2016)
                                <1>       ;cmp  al, 'Y' ; 'yes'
                                <1>       ;cmc
                                <1>         ;jnc loc_file_rw_restore_retn
000099B0 3C4E                   <1>       cmp   al, 'N' ; 'no'
000099B2 0F84C7FDFFFF           <1>         je  loc_file_rw_restore_retn
                                <1>
                                <1>       ; 29/12/2017
000099B8 E869000000             <1>       call  delete_sub_directory
000099BD 7213                   <1>       jc    short loc_rmdir_cmd_failed
                                <1>
                                <1> loc_rmdir_ok:
000099BF BE[15440100]           <1>       mov   esi, Msg_OK
000099C4 E85CDAFFFF             <1>       call  print_msg
000099C9 E9B1FDFFFF             <1>       jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_rmdir_y_n_escape:
000099CE B04E                   <1>       mov   al, 'N' ; 'no'
000099D0 EBD8                   <1>         jmp     loc_do_not_delete_directory
                                <1>
                                <1> loc_rmdir_cmd_failed:
                                <1>       ; 29/12/2017
000099D2 09C0                   <1>       or    eax, eax ; EAX = 0 -> Directory not empty!
000099D4 7426                   <1>       jz    short loc_rmdir_directory_not_empty
                                <1>
                                <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
                                <1>
000099D6 833D[D2900100]01       <1>       cmp   dword [FAT_ClusterCounter], 1
000099DD 0F829CFDFFFF           <1>       jb    loc_file_rw_cmd_failed
000099E3 F9                     <1>       stc
                                <1> loc_rmdir_cmd_return:
                                <1>       ; 01/03/2016
000099E4 9C                     <1>       pushf
                                <1>       ; ESI = Logical DOS Drive Description Table address
000099E5 66BB00FF               <1>       mov   bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
                                <1>                        ; BL = 0 -> Recalculate free cluster count
000099E9 50                     <1>       push  eax
000099EA E8C3380000             <1>       call  calculate_fat_freespace
000099EF 58                     <1>       pop   eax
000099F0 9D                     <1>       popf
000099F1 0F8288FDFFFF           <1>       jc    loc_file_rw_cmd_failed
000099F7 E983FDFFFF             <1>       jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_rmdir_directory_not_empty:
000099FC BE[3C440100]           <1>       mov   esi, Msg_Dir_Not_Empty
00009A01 E81FDAFFFF             <1>       call  print_msg
                                <1>       ; 01/03/2016
00009A06 A1[D2900100]           <1>       mov   eax, [FAT_ClusterCounter]
00009A0B 09C0                   <1>       or    eax, eax ; 0 ?
00009A0D 0F846CFDFFFF           <1>       jz    loc_file_rw_restore_retn
                                <1>       ; ESI = Logical DOS Drive Description Table address
00009A13 66BB01FF               <1>       mov   bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
                                <1>                        ; BL = 1 -> add free clusters
00009A17 E896380000             <1>       call  calculate_fat_freespace
00009A1C 09C9                   <1>       or    ecx, ecx
00009A1E 0F845BFDFFFF           <1>         jz      loc_file_rw_restore_retn ; ecx = 0 -> OK
                                <1>       ; ecx > 0 -> Error (Recalculation is needed)
00009A24 EBBE                   <1>       jmp   short loc_rmdir_cmd_return
                                <1>
                                <1>
                                <1> delete_sub_directory:
                                <1>       ; 29/12/2017
                                <1>       ; (moved here from 'delete_directory' for 'sysrmdir' )
                                <1>
                                <1>       ; EDI = Directory buffer entry offset/address
                                <1>
                                <1> loc_rmdir_delete_short_name_check_dir_empty:
00009A26 668B4714               <1>       mov   ax, [edi+20] ; First Cluster High Word
00009A2A C1E010                 <1>         shl  eax, 16
00009A2D 668B471A               <1>       mov   ax, [edi+26] ; First Cluster Low Word
                                <1>
                                <1>       ;mov  [DelFile_FCluster], eax
                                <1>
                                <1>       ;;mov bx, [DirBuff_EntryCounter]
                                <1>       ;mov  bx, [FindFile_DirEntryNumber] ; 27/02/2016
                                <1>       ;mov  [DelFile_EntryCounter], bx
```

```
6396                                    <1>
6397 00009A31 29DB                     <1>       sub    ebx, ebx
6398                                    <1>       ; 29/12/2017
6399 00009A33 891D[D2900100]            <1>       mov    [FAT_ClusterCounter], ebx ; 0 ; Reset
6400                                    <1>
6401 00009A39 8A3D[56920100]            <1>       mov    bh, [FindFile_Drv]
6402 00009A3F BE00010900                <1>       mov    esi, Logical_DOSDisks
6403 00009A44 01DE                      <1>       add    esi, ebx
6404                                    <1>
6405 00009A46 66817F0CA101              <1>       cmp    word [edi+DirEntry_NTRes], 01A1h
6406 00009A4C 745A                      <1>       je     short loc_rmdir_check_fs_directory
6407                                    <1>
6408                                    <1>       ;cmp   byte [esi+LD_FATType], 1
6409                                    <1>       ;jb    short loc_rmdir_get__last_cluster_0
6410                                    <1>
6411                                    <1>       ; 29/12/2017
6412 00009A4E 83F802                   <1>       cmp    eax, 2
6413 00009A51 7306                     <1>       jnb    short loc_rmdir_get_last_cluster_1
6414                                    <1>       ; eax < 2
6415                                    <1> loc_rmdir_get_last_cluster_0:
6416                                    <1>       ;mov   eax, ERR_INV_FORMAT ; invalid format!
6417 00009A53 B813000000               <1>       mov    eax, ERR_NOT_DIR ; not a valid directory!
6418                                    <1>       ;stc
6419 00009A58 C3                       <1>       retn
6420                                    <1>
6421                                    <1> loc_rmdir_get_last_cluster_1:
6422 00009A59 807E0303                 <1>       cmp    byte [esi+LD_FATType], 3 ; FAT32
6423 00009A5D 750C                     <1>       jne    short loc_rmdir_get_last_cluster_2
6424                                    <1>
6425                                    <1>       ; is it root directory ?
6426 00009A5F 3B4632                   <1>       cmp    eax, [esi+LD_BPB+BPB_RootClus]
6427 00009A62 7507                     <1>       jne    short loc_rmdir_get_last_cluster_2
6428                                    <1>
6429                                    <1>       ; root directory can not be deleted !!
6430                                    <1> loc_rmdir_permission_denied:
6431 00009A64 B80B000000               <1>       mov    eax, ERR_PERM_DENIED ; permission denied!
6432 00009A69 F9                       <1>       stc
6433 00009A6A C3                       <1>       retn
6434                                    <1>
6435                                    <1> loc_rmdir_get_last_cluster_2:
6436                                    <1>       ; 29/12/2017
6437 00009A6B A3[18930100]             <1>       mov    [DelFile_FCluster], eax
6438                                    <1>
6439                                    <1>       ;mov   dx, [DirBuff_EntryCounter]
6440 00009A70 668B15[D0920100]         <1>       mov    dx, [FindFile_DirEntryNumber] ; 27/02/2016
6441 00009A77 668915[1C930100]         <1>       mov    [DelFile_EntryCounter], dx
6442                                    <1>
6443 00009A7E 8B15[E1900100]           <1>       mov    edx, [DirBuff_Cluster]
6444 00009A84 8915[48930100]           <1>       mov    [RmDir_ParentDirCluster], edx
6445                                    <1>
6446 00009A8A 893D[44930100]           <1>       mov    [RmDir_DirEntryOffset], edi
6447                                    <1>
6448                                    <1>       ; 01/03/2016
6449                                    <1>       ;mov   dword [FAT_ClusterCounter], 0 ; Reset
6450                                    <1>
6451                                    <1> loc_rmdir_get_last_cluster_3:
6452 00009A90 E89C390000               <1>       call   get_last_cluster
6453                                    <1>         ;jc  loc_rmdir_cmd_failed
6454 00009A95 721E                     <1>       jc     short loc_delete_sub_dir_retn ; 29/12/2017
6455                                    <1>
6456 00009A97 3B05[18930100]           <1>       cmp    eax, [DelFile_FCluster]
6457 00009A9D 7517                     <1>       jne    short loc_rmdir_multi_dir_clusters
6458                                    <1>
6459 00009A9F C605[43930100]00         <1>       mov    byte [RmDir_MultiClusters], 0
6460 00009AA6 EB15                     <1>       jmp    short pass_rmdir_multi_dir_clusters
6461                                    <1>
6462                                    <1> loc_rmdir_check_fs_directory:
6463                                    <1>       ; 29/12/2017
6464 00009AA8 807E04A1                 <1>       cmp    byte [esi+LD_FSType], 0A1h
6465 00009AAC 75B6                     <1>       jne    short loc_rmdir_permission_denied
6466                                    <1>
6467                                    <1> loc_rmdir_delete_fs_directory:
6468 00009AAE E876130000               <1>       call   delete_fs_directory
6469                                    <1>       ;jnc   loc_print_deleted_message
6470 00009AB3 7300                     <1>       jnc    short loc_delete_sub_dir_retn ; 29/12/2017
6471                                    <1>
6472                                    <1>       ; EAX=0 -> Directory not empty !
6473                                    <1>       ; EAX>0 -> Disk r/w error or another (misc) error
6474                                    <1>
6475                                    <1>       ;or    eax, eax
6476                                    <1>       ;jz    loc_rmdir_directory_not_empty_2
6477                                    <1>       ;;stc
6478                                    <1>       ;;jmp  loc_file_rw_cmd_failed
6479                                    <1>
6480                                    <1> loc_delete_sub_dir_retn:
6481 00009AB5 C3                       <1>       retn
6482                                    <1>
6483                                    <1> loc_rmdir_multi_dir_clusters:
6484 00009AB6 C605[43930100]01         <1>       mov    byte [RmDir_MultiClusters], 1
6485                                    <1>
6486                                    <1> pass_rmdir_multi_dir_clusters:
6487 00009ABD A3[4C930100]             <1>       mov    [RmDir_DirLastCluster], eax
6488 00009AC2 890D[50930100]           <1>       mov    [RmDir_PreviousCluster], ecx
6489                                    <1>
6490                                    <1> loc_rmdir_load_fat_sub_directory:
6491 00009AC8 E84F330000               <1>       call   load_FAT_sub_directory
6492                                    <1>       ;jc    loc_rmdir_cmd_failed
6493 00009ACD 72E6                     <1>       jc     short loc_delete_sub_dir_retn
6494                                    <1>
6495                                    <1> loc_rmdir_find_last_dir_entry:
6496 00009ACF 56                       <1>       push   esi
6497 00009AD0 BE[3A920100]             <1>       mov    esi, Dir_File_Name
6498 00009AD5 C6062A                   <1>       mov    byte [esi], '*'
6499 00009AD8 C646082A                 <1>       mov    byte [esi+8], '*'
6500 00009ADC 31DB                     <1>       xor    ebx, ebx ; Entry offset  = 0
```

```
6501                                    <1> loc_rmdir_find_last_dir_entry_next:
6502 00009ADE 66B80008                  <1>         mov     ax, 0800h ; Except volume/long names
6503 00009AE2 6631C9                    <1>         xor     cx, cx ; 0 = Find a valid file or dir name
6504 00009AE5 E879170000                <1>         call    find_directory_entry
6505 00009AEA 7225                      <1>         jc      short loc_rmdir_empty_dir_cluster
6506 00009AEC 83FB01                    <1>         cmp     ebx, 1
6507 00009AEF 771B                      <1>         ja      short loc_rmdir_directory_not_empty_1
6508                                    <1> loc_rmdir_dot_entry_check:
6509 00009AF1 80FD2E                    <1>         cmp     ch, '.' ; The first char of the dir entry
6510 00009AF4 7516                      <1>         jne     short loc_rmdir_directory_not_empty_1
6511 00009AF6 08DB                      <1>         or      bl, bl
6512 00009AF8 7506                      <1>         jnz     short loc_rmdir_dotdot_entry_check
6513 00009AFA 807F0120                  <1>         cmp     byte [edi+1], 20h
6514 00009AFE EB06                      <1>         jmp     short pass_rmdir_dot_entry_check
6515                                    <1>
6516                                    <1> loc_rmdir_dotdot_entry_check:
6517 00009B00 66817F012E20              <1>         cmp     word [edi+1], '. '
6518                                    <1> pass_rmdir_dot_entry_check:
6519 00009B06 7504                      <1>         jne     short loc_rmdir_directory_not_empty_1
6520 00009B08 FEC3                      <1>         inc     bl
6521 00009B0A EBD2                      <1>         jmp     short loc_rmdir_find_last_dir_entry_next
6522                                    <1>
6523                                    <1> loc_rmdir_directory_not_empty_1:
6524 00009B0C 58                        <1>         pop     eax ; pushed esi
6525 00009B0D 31C0                      <1>         xor     eax, eax ; 0
6526                                    <1> loc_rmdir_directory_not_empty_2:
6527                                    <1> loc_delete_sub_dir_stc_retn:
6528 00009B0F F9                        <1>         stc
6529 00009B10 C3                        <1>         retn
6530                                    <1>
6531                                    <1> loc_rmdir_empty_dir_cluster:
6532 00009B11 5E                        <1>         pop     esi
6533                                    <1>
6534                                    <1> loc_rmdir_set_prev_cluster_dir_last_cluster:
6535 00009B12 803D[43930100]00          <1>         cmp     byte [RmDir_MultiClusters], 0
6536 00009B19 7613                      <1>         jna     short loc_rmdir_unlink_dir_last_cluster
6537                                    <1>
6538 00009B1B A1[50930100]              <1>         mov     eax, [RmDir_PreviousCluster]
6539                                    <1>         ;xor    ecx, ecx
6540 00009B20 49                        <1>         dec     ecx ; FFFFFFFFh
6541 00009B21 E83A340000                <1>         call    update_cluster
6542 00009B26 7306                      <1>         jnc     short loc_rmdir_unlink_dir_last_cluster
6543                                    <1>
6544                                    <1>         ; 01/03/2016
6545                                    <1>         ;cmp    eax, 1  ; eax = 0 -> end of cluster chain
6546                                    <1>         ;cmc
6547                                    <1>         ;jc     short loc_rmdir_cmd_failed
6548                                    <1>         ;jmp    short loc_rmdir_save_fat_buffer
6549                                    <1>         ; 29/12/2017
6550 00009B28 21C0                      <1>         and     eax, eax
6551 00009B2A 75E3                      <1>         jnz     short loc_delete_sub_dir_stc_retn
6552 00009B2C EB12                      <1>         jmp     short loc_rmdir_save_fat_buffer
6553                                    <1>
6554                                    <1> loc_rmdir_unlink_dir_last_cluster:
6555 00009B2E A1[4C930100]              <1>         mov     eax, [RmDir_DirLastCluster]
6556 00009B33 31C9                      <1>         xor     ecx, ecx ; 0
6557 00009B35 E826340000                <1>         call    update_cluster
6558 00009B3A 7327                      <1>         jnc     short loc_rmdir_unlink_stc_retn_0Bh
6559                                    <1>         ; Because of it is the last cluster
6560                                    <1>         ; 'update_cluster' must return with eocc error
6561 00009B3C 09C0                      <1>         or      eax, eax
6562                                    <1>         ;jz     short loc_rmdir_save_fat_buffer ; eocc
6563                                    <1>         ;stc
6564                                    <1>           ;jmp      short loc_rmdir_cmd_failed
6565                                    <1>         ; 29/12/2017
6566 00009B3E 75CF                      <1>         jnz     short loc_delete_sub_dir_stc_retn
6567                                    <1>
6568                                    <1> loc_rmdir_save_fat_buffer:
6569 00009B40 803D[CA900100]02          <1>         cmp     byte [FAT_BuffValidData], 2
6570 00009B47 7528                      <1>         jne     short loc_rmdir_calculate_FAT_freespace
6571 00009B49 E8CF360000                <1>         call    save_fat_buffer
6572                                    <1>         ;jc     short loc_rmdir_cmd_failed
6573                                    <1>         ; 29/12/2017
6574 00009B4E 7219                      <1>         jc      short loc_rmdir_unlink_error_retn
6575                                    <1>
6576                                    <1>         ; 01/03/2016
6577 00009B50 803D[43930100]00          <1>         cmp     byte [RmDir_MultiClusters], 0
6578 00009B57 7618                      <1>         jna     short loc_rmdir_calculate_FAT_freespace
6579                                    <1>
6580 00009B59 A1[18930100]              <1>         mov     eax, [DelFile_FCluster]
6581 00009B5E E92DFFFFFF                <1>           jmp     loc_rmdir_get_last_cluster_3
6582                                    <1>
6583                                    <1> loc_rmdir_unlink_stc_retn_0Bh:
6584                                    <1>         ; 15/10/2016 (0Bh -> 28)
6585 00009B63 B81C000000                <1>         mov     eax, ERR_INV_FORMAT ; 28 = Invalid format
6586                                    <1> loc_rmdir_unlink_stc_retn:
6587 00009B68 F9                        <1>         stc
6588                                    <1> loc_rmdir_unlink_error_retn:
6589 00009B69 C3                        <1>         retn
6590                                    <1>
6591                                    <1> loc_rmdir_delete_short_name_invalid_data:
6592 00009B6A B81D000000                <1>         mov     eax, 29 ; Invalid data (15/10/2016)
6593                                    <1>         ;stc
6594                                    <1>           ;jmp loc_rmdir_cmd_failed
6595                                    <1>         ; 29/12/2017
6596 00009B6F EBF7                      <1>         jmp     short loc_rmdir_unlink_stc_retn
6597                                    <1>
6598                                    <1> loc_rmdir_calculate_FAT_freespace:
6599                                    <1>         ;mov    eax, [FAT_ClusterCounter]
6600                                    <1>         ; 29/12/2017
6601 00009B71 29C0                      <1>         sub     eax, eax ; 0
6602 00009B73 8705[D2900100]            <1>         xchg    eax, [FAT_ClusterCounter]
6603                                    <1>         ;
6604 00009B79 66BB01FF                  <1>         mov     bx, 0FF01h
6605                                    <1>         ; BL = 1 -> Add EAX to free space count
```

```
6606                                <1>       ; BH = FFh ->
6607                                <1>       ; ESI = Logical DOS Drive Description Table address
6608 00009B7D E830370000            <1>       call   calculate_fat_freespace
6609                                <1>
6610 00009B82 21C9                  <1>       and    ecx, ecx ; ecx = 0 -> valid free sector count
6611 00009B84 7409                  <1>       jz     short loc_rmdir_delete_short_name_continue
6612                                <1>
6613                                <1> loc_rmdir_recalculate_FAT_freespace:
6614 00009B86 66BB00FF              <1>         mov     bx, 0FF00h ; BL = 0 -> Recalculate free space
6615 00009B8A E823370000            <1>       call   calculate_fat_freespace
6616                                <1>
6617                                <1> loc_rmdir_delete_short_name_continue:
6618 00009B8F A1[48930100]          <1>       mov    eax, [RmDir_ParentDirCluster]
6619 00009B94 83F802                <1>       cmp    eax, 2
6620 00009B97 7309                  <1>       jnb    short loc_rmdir_del_short_name_load_sub_dir
6621 00009B99 E8F3310000            <1>       call   load_FAT_root_directory
6622                                <1>       ;jc    loc_file_rw_cmd_failed
6623                                <1>       ; 29/12/2017
6624 00009B9E 72C9                  <1>       jc     short loc_rmdir_unlink_error_retn
6625 00009BA0 EB07                  <1>       jmp    short loc_rmdir_del_short_name_ld_chk_fclust
6626                                <1>
6627                                <1> loc_rmdir_del_short_name_load_sub_dir:
6628 00009BA2 E875320000            <1>       call   load_FAT_sub_directory
6629                                <1>       ;jc    loc_file_rw_cmd_failed
6630                                <1>       ; 29/12/2017
6631 00009BA7 72C0                  <1>       jc     short loc_rmdir_unlink_error_retn
6632                                <1>
6633                                <1> loc_rmdir_del_short_name_ld_chk_fclust:
6634 00009BA9 0FB73D[44930100]      <1>       movzx  edi, word [RmDir_DirEntryOffset]
6635 00009BB0 81C700000800          <1>       add    edi, Directory_Buffer
6636                                <1>
6637 00009BB6 668B4714              <1>       mov    ax, [edi+20] ; First Cluster High Word
6638 00009BBA C1E010                <1>       shl    eax, 16
6639 00009BBD 668B471A              <1>       mov    ax, [edi+26] ; First Cluster Low Word
6640                                <1>         ; Not necessary...
6641 00009BC1 3B05[18930100]        <1>       cmp    eax, [DelFile_FCluster]
6642 00009BC7 75A1                  <1>       jne    short loc_rmdir_delete_short_name_invalid_data
6643                                <1>       ;
6644 00009BC9 C607E5                <1>       mov    byte [edi], 0E5h ; 'Deleted' sign
6645                                <1>       ; 27/02/2016
6646                                <1>       ; TRDOS v1 has a bug here! it does not set
6647                                <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
6648                                <1>       ; 'save_directory_buffer' would not save the change !
6649 00009BCC C605[DC900100]02      <1>       mov    byte [DirBuff_ValidData], 2 ; change sign
6650                                <1>       ;
6651 00009BD3 E8AE1D0000            <1>       call   save_directory_buffer
6652                                <1>       ;jc    loc_file_rw_cmd_failed
6653                                <1>       ; 29/12/2017
6654 00009BD8 728F                  <1>       jc     short loc_rmdir_unlink_error_retn
6655                                <1>
6656                                <1> loc_rmdir_del_long_name:
6657 00009BDA 0FB615[1E930100]      <1>       movzx  edx, byte [DelFile_LNEL]
6658 00009BE1 08D2                  <1>       or     dl, dl
6659 00009BE3 7410                  <1>       jz     short loc_rmdir_update_parent_dir_lmdt
6660                                <1>
6661 00009BE5 0FB705[1C930100]      <1>       movzx  eax, word [DelFile_EntryCounter]
6662 00009BEC 29D0                  <1>       sub    eax, edx
6663                                <1>       ; 29/12/2017
6664 00009BEE 7205                  <1>       jc     short loc_rmdir_update_parent_dir_lmdt
6665                                <1>
6666                                <1>       ; EAX = Directory Entry Number of the long name last entry
6667 00009BF0 E8EF1E0000            <1>       call   delete_longname
6668                                <1>
6669                                <1> loc_rmdir_update_parent_dir_lmdt:
6670 00009BF5 E8271E0000            <1>       call   update_parent_dir_lmdt
6671                                <1>       ;jc    short loc_file_rw_cmd_failed
6672                                <1>       ; 29/12/2017
6673                                <1>       ;jc    short loc_rmdir_unlink_error_retn
6674                                <1>
6675                                <1> loc_delete_sub_directory_ok:
6676                                <1>       ; 29/12/2017
6677 00009BFA 31C0                  <1>       xor    eax, eax ;  0 ;  cf = 0
6678 00009BFC C3                    <1>       retn
6679                                <1>
6680                                <1>
6681                                <1> delete_file:
6682                                <1>       ; 29/02/2016
6683                                <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
6684                                <1>       ; 09/08/2010 (CMD_INTR.ASM, 'cmp_cmd_del')
6685                                <1>       ; 28/02/2010
6686                                <1>
6687                                <1> get_delfile_fchar:
6688                                <1>       ; esi = file name
6689 00009BFD 803E20                <1>       cmp    byte [esi], 20h
6690 00009C00 7701                  <1>         ja   short loc_delfile_parse_path_name
6691                                <1>
6692                                <1> loc_delfile_nofilename_retn:
6693 00009C02 C3                    <1>       retn
6694                                <1>
6695                                <1> loc_delfile_parse_path_name:
6696 00009C03 BF[56920100]          <1>       mov    edi, FindFile_Drv
6697 00009C08 E815190000            <1>       call   parse_path_name
6698 00009C0D 0F822EF2FFFF          <1>       jc     loc_cmd_failed
6699                                <1>
6700                                <1> loc_delfile_check_filename_exists:
6701 00009C13 BE[98920100]          <1>       mov    esi, FindFile_Name
6702 00009C18 803E20                <1>       cmp    byte [esi], 20h
6703 00009C1B 0F8620F2FFFF          <1>       jna    loc_cmd_failed
6704 00009C21 8935[14930100]        <1>       mov    [DelFile_FNPointer], esi
6705                                <1>
6706                                <1> loc_delfile_drv:
6707 00009C27 8A15[56920100]        <1>       mov    dl, [FindFile_Drv]
6708 00009C2D 8A35[B6890100]        <1>       mov    dh, [Current_Drv]
6709 00009C33 8835[12910100]        <1>       mov    [RUN_CDRV], dh
6710 00009C39 38F2                  <1>       cmp    dl, dh
```

```
6711 00009C3B 740B                <1>       je    short loc_delfile_change_directory
6712                              <1>
6713 00009C3D E86CE3FFFF          <1>       call  change_current_drive
6714 00009C42 0F8237FBFFFF        <1>       jc    loc_file_rw_cmd_failed
6715                              <1>
6716                              <1> loc_delfile_change_directory:
6717 00009C48 803D[57920100]20    <1>       cmp   byte [FindFile_Directory], 20h
6718 00009C4F 7618                <1>       jna   short loc_delfile_find
6719                              <1>
6720 00009C51 FE05[833F0100]      <1>       inc   byte [Restore_CDIR]
6721 00009C57 BE[57920100]        <1>       mov   esi, FindFile_Directory
6722 00009C5C 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
6723 00009C5E E8A9120000          <1>       call  change_current_directory
6724 00009C63 0F8216FBFFFF        <1>       jc    loc_file_rw_cmd_failed
6725                              <1>
6726                              <1> ;loc_delfile_change_prompt_dir_string:
6727                              <1>       ;call  change_prompt_dir_string
6728                              <1>
6729                              <1> loc_delfile_find:
6730                              <1>       ;mov   esi, FindFile_Name
6731 00009C69 8B35[14930100]      <1>       mov   esi, [DelFile_FNPointer]
6732 00009C6F 66B80018            <1>       mov   ax, 1800h ; Except volume label and dirs
6733 00009C73 E8D9F6FFFF          <1>       call  find_first_file
6734 00009C78 0F8201FBFFFF        <1>       jc    loc_file_rw_cmd_failed
6735                              <1>
6736                              <1> loc_delfile_ambgfn_check:
6737 00009C7E 6621D2              <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
6738 00009C81 740B                <1>       jz    short loc_delfile_found
6739                              <1>
6740                              <1> loc_file_not_found:
6741 00009C83 B802000000          <1>       mov   eax, 2 ; File not found sign
6742 00009C88 F9                  <1>       stc
6743 00009C89 E9F1FAFFFF          <1>       jmp   loc_file_rw_cmd_failed
6744                              <1>
6745                              <1> loc_delfile_found:
6746 00009C8E 80E307              <1>       and   bl, 07h ; Attributes
6747 00009C91 0F85F2FAFFFF        <1>         jnz     loc_permission_denied
6748                              <1>
6749                              <1> ;loc_delfile_found_save_lnel:
6750                              <1> ;     mov   [DelFile_LNEL], bh ; Long name entry length (if > 0)
6751                              <1>
6752                              <1> loc_delfile_ask_for_delete:
6753 00009C97 57                  <1>       push  edi ; * (29/02/2016)
6754                              <1>
6755 00009C98 BE[53440100]        <1>       mov   esi, Msg_DoYouWantDelete
6756 00009C9D E883D7FFFF          <1>       call  print_msg
6757 00009CA2 8B35[14930100]      <1>       mov   esi, [DelFile_FNPointer]
6758 00009CA8 E878D7FFFF          <1>       call  print_msg
6759 00009CAD BE[07440100]        <1>       mov   esi, Msg_YesNo
6760 00009CB2 E86ED7FFFF          <1>       call  print_msg
6761                              <1>
6762                              <1> loc_delfile_ask_again:
6763 00009CB7 30E4                <1>       xor   ah, ah
6764 00009CB9 E82572FFFF          <1>       call  int16h
6765 00009CBE 3C1B                <1>       cmp   al, 1Bh
6766                              <1>       ;je   short loc_do_not_delete_file
6767 00009CC0 7449                <1>       je    short loc_delfile_y_n_escape ; 06/03/2016
6768 00009CC2 24DF                <1>       and   al, 0DFh
6769 00009CC4 A2[11440100]        <1>       mov   [Y_N_nextline], al
6770 00009CC9 3C59                <1>       cmp   al, 'Y'
6771 00009CCB 7404                <1>       je    short loc_yes_delete_file
6772 00009CCD 3C4E                <1>       cmp   al, 'N'
6773 00009CCF 75E6                <1>       jne   short loc_delfile_ask_again
6774                              <1>
6775                              <1> loc_do_not_delete_file:
6776                              <1> loc_yes_delete_file:
6777 00009CD1 E8E2FBFFFF          <1>       call  y_n_answer ; 29/12/2017
6778 00009CD6 5F                  <1>       pop   edi ; * (29/02/2016)
6779                              <1>       ;cmp  al, 'Y' ; 'yes'
6780                              <1>       ;cmc
6781                              <1>         ;jnc loc_file_rw_restore_retn
6782 00009CD7 3C4E                <1>       cmp   al, 'N' ; 'no'
6783 00009CD9 0F84A0FAFFFF        <1>         je  loc_file_rw_restore_retn
6784                              <1>
6785                              <1> loc_delete_file:
6786 00009CDF 8A3D[56920100]      <1>       mov   bh, [FindFile_Drv]
6787                              <1>       ;mov  bl, [DelFile_LNEL]
6788 00009CE5 8A1D[A5920100]      <1>       mov   bl, [FindFile_LongNameEntryLength]
6789                              <1>       ;mov  cx, [DirBuff_EntryCounter]
6790 00009CEB 668B0D[D0920100]    <1>       mov   cx, [FindFile_DirEntryNumber]
6791                              <1>       ; (*) EDI = Directory buffer entry offset/address
6792 00009CF2 E8D71F0000          <1>       call  remove_file ; (FILE.ASM, 'proc_delete_file')
6793 00009CF7 0F8378FAFFFF        <1>       jnc   loc_print_deleted_message
6794                              <1>
6795                              <1>       ;cmp  al, 05h
6796 00009CFD 3C0B                <1>       cmp   al, ERR_PERM_DENIED  ; 29/12/2017 (5 -> 11)
6797 00009CFF 0F8484FAFFFF        <1>       je    loc_permission_denied
6798 00009D05 F9                  <1>       stc
6799 00009D06 E974FAFFFF          <1>       jmp   loc_file_rw_cmd_failed
6800                              <1>
6801                              <1> loc_delfile_y_n_escape:
6802 00009D0B B04E                <1>       mov   al, 'N' ; 'no'
6803 00009D0D EBC2                <1>       jmp   short loc_do_not_delete_file
6804                              <1>
6805                              <1> set_file_attributes:
6806                              <1>       ; 06/03/2016
6807                              <1>       ; 04/03/2016 (TRDOS 386 = TRDOS v2.0)
6808                              <1>       ; 10/07/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_attrib')
6809                              <1>       ; 23/05/2010
6810                              <1>       ; 17/12/2000 (P2000.ASM)
6811                              <1>
6812                              <1>       ; esi = file or directory name
6813 00009D0F 6631C0              <1>       xor   ax, ax
6814 00009D12 66A3[A4440100]      <1>       mov   [Attr_Chars], ax
6815 00009D18 A2[6C930100]        <1>       mov   [Attributes], al
```

```
                                <1>
                                <1> get_attrib_fchar:
                                <1>       ; esi = file name
00009D1D 8A06                   <1>       mov    al, [esi]
00009D1F 3C20                   <1>       cmp    al, 20h
00009D21 7623                   <1>       jna    short loc_attr_file_nofilename_retn
                                <1>
                                <1> loc_scan_attrib_params:
00009D23 3C2D                   <1>       cmp    al, '-'
00009D25 0F871C010000           <1>       ja     loc_attr_file_parse_path_name
00009D2B 7408                   <1>       je     short loc_attr_space
                                <1>
00009D2D 3C2B                   <1>       cmp    al, '+'
00009D2F 0F850CF1FFFF           <1>       jne    loc_cmd_failed
                                <1>
                                <1> loc_attr_space:
00009D35 8A6601                 <1>       mov    ah, [esi+1]
00009D38 80FC20                 <1>       cmp    ah, 20h
00009D3B 770A                   <1>       ja     short pass_attr_space
00009D3D 0F82FEF0FFFF           <1>       jb     loc_cmd_failed
00009D43 46                     <1>       inc    esi
00009D44 EBEF                   <1>       jmp    short loc_attr_space
                                <1>
                                <1> loc_attr_file_nofilename_retn:
00009D46 C3                     <1>       retn
                                <1>
                                <1> pass_attr_space:
00009D47 80E4DF                 <1>       and    ah, 0DFh
00009D4A 80FC53                 <1>       cmp    ah, 'S'
00009D4D 0F87EEF0FFFF           <1>       ja     loc_cmd_failed
00009D53 7204                   <1>       jb     short pass_attr_system
00009D55 B404                   <1>       mov    ah, 04h   ; System
00009D57 EB21                   <1>       jmp    short pass_attr_archive
                                <1>
                                <1> pass_attr_system:
00009D59 80FC48                 <1>       cmp    ah, 'H'
00009D5C 7706                   <1>       ja     short pass_attr_hidden
00009D5E 7213                   <1>       jb     short pass_attr_read_only
00009D60 B402                   <1>       mov    ah, 02h    ; Hidden
00009D62 EB16                   <1>       jmp    short pass_attr_archive
                                <1>
                                <1> pass_attr_hidden:
00009D64 80FC52                 <1>       cmp    ah, 'R'
00009D67 0F87D4F0FFFF           <1>       ja     loc_cmd_failed
00009D6D 7204                   <1>       jb     short pass_attr_read_only ; Read only
00009D6F B401                   <1>       mov    ah, 01h
00009D71 EB07                   <1>       jmp    short pass_attr_archive
                                <1>
                                <1> pass_attr_read_only:
00009D73 80FC41                 <1>       cmp    ah, 'A'
00009D76 753B                   <1>       jne    short loc_chk_attr_enter
00009D78 B420                   <1>       mov    ah, 20h    ; Archive
                                <1>
                                <1> pass_attr_archive:
00009D7A 3C2D                   <1>       cmp    al, '-'
00009D7C 7508                   <1>       jne    short pass_reducing_attributes
00009D7E 0825[A4440100]         <1>       or     [Attr_Chars], ah
00009D84 EB06                   <1>       jmp    short loc_change_attributes_inc
                                <1>
                                <1> pass_reducing_attributes:
00009D86 0825[A5440100]         <1>       or     [Attr_Chars+1], ah
                                <1>
                                <1> loc_change_attributes_inc:
00009D8C 46                     <1>       inc    esi
00009D8D 8A6601                 <1>       mov    ah, [esi+1]
00009D90 80FC20                 <1>       cmp    ah, 20h
00009D93 7227                   <1>       jb     short pass_change_attr
00009D95 74F5                   <1>       je     short loc_change_attributes_inc
00009D97 80FC2D                 <1>       cmp    ah, '-'
00009D9A 770D                   <1>       ja     short loc_chk_next_attr_char1
00009D9C 7405                   <1>       je     short loc_chk_next_attr_char0
00009D9E 80FC2B                 <1>       cmp    ah, '+'
00009DA1 7506                   <1>       jne    short loc_chk_next_attr_char1
                                <1>
                                <1> loc_chk_next_attr_char0:
00009DA3 46                     <1>       inc    esi
00009DA4 668B06                 <1>       mov    ax, [esi]
00009DA7 EB9E                   <1>       jmp    short pass_attr_space
                                <1>
                                <1> loc_chk_next_attr_char1:
00009DA9 803E2D                 <1>       cmp    byte [esi], '-'
00009DAC 7799                   <1>       ja     short pass_attr_space
00009DAE E988000000             <1>         jmp     loc_attr_file_check_fname_fchar
                                <1>
                                <1> loc_chk_attr_enter:
00009DB3 80FC0D                 <1>       cmp    ah, 0Dh
00009DB6 0F8585F0FFFF           <1>       jne    loc_cmd_failed
                                <1>
                                <1> pass_change_attr:
00009DBC A0[A4440100]           <1>       mov    al, [Attr_Chars]
00009DC1 F6D0                   <1>       not    al
00009DC3 2005[6C930100]         <1>       and    [Attributes], al
00009DC9 A0[A5440100]           <1>       mov    al, [Attr_Chars+1]
00009DCE 0805[6C930100]         <1>       or     [Attributes], al
                                <1>
                                <1> loc_show_attributes:
00009DD4 BE[1F4C0100]           <1>       mov    esi, nextline
00009DD9 E847D6FFFF             <1>       call   print_msg
                                <1>
                                <1> loc_show_attributes_no_nextline:
00009DDE C705[A4440100]4E4F-    <1>       mov    dword [Attr_Chars], 'NORM'
00009DE6 524D                   <1>
00009DE8 66C705[A8440100]41-    <1>       mov    word [Attr_Chars+4], 'AL'
00009DF0 4C                     <1>
00009DF1 BE[A4440100]           <1>       mov    esi, Attr_Chars
```

```
6919 00009DF6 A0[6C930100]        <1>       mov   al, [Attributes]
6920 00009DFB A804                <1>       test  al, 04h
6921 00009DFD 7406                <1>       jz    short pass_put_attr_s
6922 00009DFF 66C7065300          <1>       mov   word [esi], 0053h     ; S
6923 00009E04 46                  <1>       inc   esi
6924                              <1>
6925                              <1> pass_put_attr_s:
6926 00009E05 A802                <1>       test  al, 02h
6927 00009E07 7406                <1>       jz    short pass_put_attr_h
6928 00009E09 66C7064800          <1>       mov   word [esi], 0048h     ; H
6929 00009E0E 46                  <1>       inc   esi
6930                              <1>
6931                              <1> pass_put_attr_h:
6932 00009E0F A801                <1>       test  al, 01h
6933 00009E11 7406                <1>       jz    short pass_put_attr_r
6934 00009E13 66C7065200          <1>       mov   word [esi], 0052h     ; R
6935 00009E18 46                  <1>       inc   esi
6936                              <1>
6937                              <1> pass_put_attr_r:
6938 00009E19 3C20                <1>       cmp   al, 20h
6939 00009E1B 7205                <1>       jb    short pass_put_attr_a
6940 00009E1D 66C7064100          <1>       mov   word [esi], 0041h     ; A
6941                              <1>
6942                              <1> pass_put_attr_a:
6943 00009E22 BE[97440100]        <1>       mov   esi, Str_Attributes
6944 00009E27 E8F9D5FFFF          <1>       call  print_msg
6945 00009E2C BE[1F4C0100]        <1>       mov   esi, nextline
6946 00009E31 E8EFD5FFFF          <1>       call  print_msg
6947 00009E36 E944F9FFFF          <1>       jmp   loc_file_rw_restore_retn
6948                              <1>
6949                              <1> loc_attr_file_check_fname_fchar:
6950 00009E3B 46                  <1>       inc   esi
6951 00009E3C 803E20              <1>       cmp   byte [esi], 20h
6952 00009E3F 74FA                <1>       je    short loc_attr_file_check_fname_fchar
6953 00009E41 0F8275FFFFFF        <1>         jb      pass_change_attr
6954                              <1>
6955                              <1> loc_attr_file_parse_path_name:
6956 00009E47 BF[56920100]        <1>       mov   edi, FindFile_Drv
6957 00009E4C E8D1160000          <1>       call  parse_path_name
6958 00009E51 0F82EAEFFFFF        <1>       jc    loc_cmd_failed
6959                              <1>
6960                              <1> loc_attr_file_check_filename_exists:
6961 00009E57 BE[98920100]        <1>       mov   esi, FindFile_Name
6962 00009E5C 803E20              <1>       cmp   byte [esi], 20h
6963 00009E5F 0F86DCEFFFFF        <1>       jna   loc_cmd_failed
6964 00009E65 8935[14930100]      <1>       mov   [DelFile_FNPointer], esi
6965                              <1>
6966                              <1> loc_attr_file_drv:
6967 00009E6B 8A35[B6890100]      <1>       mov   dh, [Current_Drv]
6968 00009E71 8835[12910100]      <1>       mov   [RUN_CDRV], dh
6969                              <1>
6970 00009E77 8A15[56920100]      <1>       mov   dl, [FindFile_Drv]
6971 00009E7D 38F2                <1>       cmp   dl, dh
6972 00009E7F 740B                <1>       je    short loc_attr_file_change_directory
6973                              <1>
6974 00009E81 E828E1FFFF          <1>       call  change_current_drive
6975 00009E86 0F82F3F8FFFF        <1>       jc    loc_file_rw_cmd_failed
6976                              <1>
6977                              <1> loc_attr_file_change_directory:
6978 00009E8C 803D[57920100]20    <1>         cmp     byte [FindFile_Directory], 20h
6979 00009E93 7618                <1>       jna   short loc_attr_file_find
6980                              <1>
6981 00009E95 FE05[833F0100]      <1>       inc   byte [Restore_CDIR]
6982                              <1>
6983 00009E9B BE[57920100]        <1>       mov   esi, FindFile_Directory
6984 00009EA0 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
6985 00009EA2 E865100000          <1>       call  change_current_directory
6986 00009EA7 0F82D2F8FFFF        <1>       jc    loc_file_rw_cmd_failed
6987                              <1>
6988                              <1> ;loc_attr_file_change_prompt_dir_string:
6989                              <1>       ;call  change_prompt_dir_string
6990                              <1>
6991                              <1> loc_attr_file_find:
6992                              <1>       ;mov   esi, FindFile_Name
6993 00009EAD 8B35[14930100]      <1>       mov   esi, [DelFile_FNPointer]
6994 00009EB3 66B80008            <1>       mov   ax, 0800h ; Except volume labels
6995 00009EB7 E895F4FFFF          <1>       call  find_first_file
6996 00009EBC 0F82BDF8FFFF        <1>       jc    loc_file_rw_cmd_failed
6997                              <1>
6998                              <1> loc_attr_file_ambgfn_check:
6999 00009EC2 6609D2              <1>       or    dx, dx ; Ambiguous filename chars used sign (DX>0)
7000                              <1>       ;     (Note: It was BX in TRDOS v1)
7001                              <1>       ;jz   short loc_attr_file_found
7002 00009EC5 0F85B8FDFFFF        <1>         jnz     loc_file_not_found ; 06/03/2016
7003                              <1>
7004                              <1>       ;mov  eax, 2 ; File not found sign
7005                              <1>       ;stc
7006                              <1>       ;jmp  loc_file_rw_cmd_failed
7007                              <1>
7008                              <1> loc_attr_file_found:
7009                              <1>       ; EDI = Directory buffer entry offset/address
7010                              <1>       ; BL = File (or Directory) Attributes
7011                              <1>       ;     (Note: It was 'CL' in TRDOS v1)
7012                              <1>       ; mov  bl, [EDI+0Bh]
7013                              <1>
7014 00009ECB 66833D[A4440100]00  <1>       cmp   word [Attr_Chars], 0
7015 00009ED3 770B                <1>       ja    short loc_attr_file_change_attributes
7016 00009ED5 881D[6C930100]      <1>       mov   [Attributes], bl
7017 00009EDB E9F4FEFFFF          <1>       jmp   loc_show_attributes
7018                              <1>
7019                              <1> loc_attr_file_change_attributes:
7020 00009EE0 A0[A4440100]        <1>       mov   al, [Attr_Chars]
7021 00009EE5 F6D0                <1>       not   al
7022 00009EE7 20C3                <1>       and   bl, al
7023 00009EE9 A0[A5440100]        <1>       mov   al, [Attr_Chars+1]
```

```
7024 00009EEE 08C3                <1>       or    bl, al
7025                              <1>
7026 00009EF0 66817F0CA101        <1>       cmp   word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
7027 00009EF6 741D                <1>       je    short loc_attr_file_fs_check
7028                              <1>
7029 00009EF8 881D[6C930100]      <1>       mov   [Attributes], bl
7030 00009EFE 885F0B              <1>       mov   [edi+0Bh], bl    ; Attributes (New!)
7031                              <1>
7032                              <1>       ; 04/03/2016
7033                              <1>       ; TRDOS v1 has a bug here! it does not set
7034                              <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
7035                              <1>       ; 'save_directory_buffer' would not save the new attributes !
7036                              <1>
7037 00009F01 C605[DC900100]02    <1>       mov   byte [DirBuff_ValidData], 2
7038                              <1>
7039 00009F08 E8791A0000          <1>       call  save_directory_buffer
7040 00009F0D 0F826CF8FFFF        <1>       jc    loc_file_rw_cmd_failed
7041                              <1>
7042 00009F13 EB33                <1>       jmp   short loc_print_attr_changed_message
7043                              <1>
7044                              <1> loc_attr_file_fs_check:
7045 00009F15 29C0                <1>       sub   eax, eax
7046 00009F17 8A25[DA900100]      <1>         mov     ah, [DirBuff_DRV]
7047 00009F1D BE00010900          <1>       mov   esi, Logical_DOSDisks
7048 00009F22 01C6                <1>         add     esi, eax
7049 00009F24 807E04A1            <1>         cmp     byte [esi+LD_FSType], 0A1h
7050 00009F28 7309                <1>       jnc   short loc_attr_file_change_fs_file_attributes
7051                              <1>       ; 29/12/2017 (0Dh -> 29)
7052 00009F2A 66B81D00            <1>       mov   ax, 29 ; Invalid Data
7053 00009F2E E94CF8FFFF          <1>       jmp   loc_file_rw_cmd_failed
7054                              <1>
7055                              <1> loc_attr_file_change_fs_file_attributes:
7056                              <1>       ; BL = New MS-DOS File Attributes
7057 00009F33 88D8                <1>       mov   al, bl ; File/Directory Attributes
7058 00009F35 30E4                <1>       xor   ah, ah ; Attributes in MS-DOS format sign
7059 00009F37 E873050000          <1>       call  change_fs_file_attributes
7060 00009F3C 0F823DF8FFFF        <1>       jc    loc_file_rw_cmd_failed
7061                              <1>
7062 00009F42 881D[6C930100]      <1>       mov   [Attributes], bl
7063                              <1>
7064                              <1> loc_print_attr_changed_message:
7065 00009F48 BE[92440100]        <1>       mov   esi, Msg_New
7066 00009F4D E8D3D4FFFF          <1>       call  print_msg
7067 00009F52 E987FEFFFF          <1>       jmp   loc_show_attributes_no_nextline
7068                              <1>
7069                              <1> rename_file:
7070                              <1>       ; 13/11/2017
7071                              <1>       ; 06/11/2016
7072                              <1>       ; 05/11/2016
7073                              <1>       ; 16/10/2016
7074                              <1>       ; 08/03/2016
7075                              <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
7076                              <1>       ; 20/11/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_rename')
7077                              <1>       ; 16/11/2010
7078                              <1>
7079                              <1> get_rename_source_fchar:
7080                              <1>       ; esi = file name
7081 00009F57 803E20              <1>       cmp   byte [esi], 20h
7082 00009F5A 7614                <1>         jna  short loc_rename_nofilename_retn
7083                              <1>
7084 00009F5C 8935[94930100]      <1>       mov   [SourceFilePath], esi
7085                              <1>
7086                              <1> rename_scan_source_file:
7087 00009F62 46                  <1>       inc   esi
7088 00009F63 803E20              <1>       cmp   byte [esi], 20h
7089 00009F66 7409                <1>       je    short rename_scan_destination_file_1
7090                              <1>       ;jb   short loc_rename_nofilename_retn
7091 00009F68 0F82D3EEFFFF        <1>       jb    loc_cmd_failed
7092 00009F6E EBF2                <1>       jmp   short rename_scan_source_file
7093                              <1>
7094                              <1> loc_rename_nofilename_retn: ; 08/03/2016
7095 00009F70 C3                  <1>       retn
7096                              <1>
7097                              <1> rename_scan_destination_file_1:
7098 00009F71 C60600              <1>       mov   byte [esi], 0
7099                              <1>
7100                              <1> rename_scan_destination_file_2:
7101 00009F74 46                  <1>       inc   esi
7102 00009F75 803E20              <1>       cmp   byte [esi], 20h
7103 00009F78 74FA                <1>       je    short rename_scan_destination_file_2
7104                              <1>       ;jb   short loc_rename_nofilename_retn
7105 00009F7A 0F82C1EEFFFF        <1>       jb    loc_cmd_failed
7106                              <1>
7107 00009F80 8935[98930100]      <1>       mov   [DestinationFilePath], esi
7108                              <1>
7109                              <1> rename_scan_destination_file_3:
7110 00009F86 46                  <1>       inc   esi
7111 00009F87 803E20              <1>       cmp   byte [esi], 20h
7112 00009F8A 77FA                <1>       ja    short rename_scan_destination_file_3
7113                              <1>
7114 00009F8C C60600              <1>       mov   byte [esi], 0
7115                              <1>
7116                              <1> loc_rename_save_current_drive:
7117 00009F8F 8A35[B6890100]      <1>       mov   dh, [Current_Drv]
7118 00009F95 8835[12910100]      <1>       mov   byte [RUN_CDRV], dh
7119                              <1>
7120                              <1> loc_rename_sf_parse_path_name:
7121 00009F9B 8B35[94930100]      <1>       mov   esi, [SourceFilePath]
7122 00009FA1 BF[56920100]        <1>       mov   edi, FindFile_Drv
7123 00009FA6 E877150000          <1>       call  parse_path_name
7124 00009FAB 0F8290EEFFFF        <1>       jc    loc_cmd_failed
7125                              <1>
7126                              <1> loc_rename_sf_check_filename_exists:
7127 00009FB1 BE[98920100]        <1>       mov   esi, FindFile_Name
7128 00009FB6 803E20              <1>       cmp   byte [esi], 20h
```

```
7129 00009FB9 0F8682EEFFFF        <1>       jna    loc_cmd_failed
7130                              <1>
7131                              <1>       ;mov   [DelFile_FNPointer], esi
7132                              <1>
7133                              <1> loc_rename_sf_drv:
7134                              <1>       ;mov   dh, [Current_Drv]
7135                              <1>       ;mov   [RUN_CDRV], dh
7136                              <1>
7137 00009FBF 8A15[56920100]      <1>       mov    dl, [FindFile_Drv]
7138 00009FC5 38F2                <1>       cmp    dl, dh ; dh = [Current_Drv]
7139 00009FC7 740B                <1>       je     short rename_sf_change_directory
7140                              <1>
7141 00009FC9 E8E0DFFFFF          <1>       call   change_current_drive
7142 00009FCE 0F82ABF7FFFF        <1>       jc     loc_file_rw_cmd_failed
7143                              <1>
7144                              <1> rename_sf_change_directory:
7145 00009FD4 803D[57920100]20    <1>       cmp    byte [FindFile_Directory], 20h
7146 00009FDB 7618                <1>       jna    short rename_sf_find
7147                              <1>
7148 00009FDD FE05[833F0100]      <1>       inc    byte [Restore_CDIR]
7149 00009FE3 BE[57920100]        <1>       mov    esi, FindFile_Directory
7150 00009FE8 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
7151 00009FEA E81D0F0000          <1>       call   change_current_directory
7152 00009FEF 0F828AF7FFFF        <1>       jc     loc_file_rw_cmd_failed
7153                              <1>
7154                              <1> ;rename_sf_change_prompt_dir_string:
7155                              <1>       ;call  change_prompt_dir_string
7156                              <1>
7157                              <1> rename_sf_find:
7158                              <1>       ;mov   esi, [DelFile_FNPointer]
7159 00009FF5 BE[98920100]        <1>       mov    esi, FindFile_Name
7160                              <1>
7161 00009FFA 66B80008            <1>       mov    ax, 0800h ; Except volume labels
7162 00009FFE E84EF3FFFF          <1>       call   find_first_file
7163 0000A003 0F8276F7FFFF        <1>       jc     loc_file_rw_cmd_failed
7164                              <1>
7165                              <1> loc_rename_sf_ambgfn_check:
7166 0000A009 6621D2              <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
7167                              <1>       ;      (Note: It was BX in TRDOS v1)
7168                              <1>       ;jz    short loc_rename_sf_found
7169 0000A00C 0F8571FCFFFF        <1>       jnz    loc_file_not_found
7170                              <1>
7171                              <1>       ;mov   eax, 2 ; File not found sign
7172                              <1>       ;stc
7173                              <1>       ;jmp   loc_file_rw_cmd_failed
7174                              <1>
7175                              <1> loc_rename_sf_found:
7176                              <1>       ; EDI = Directory buffer entry offset/address
7177                              <1>       ; BL = File (or Directory) Attributes
7178                              <1>       ;      (Note: It was 'CL' in TRDOS v1)
7179                              <1>       ; mov  bl, [EDI+0Bh]
7180                              <1>
7181 0000A012 F6C307              <1>       test   bl, 07h ; Attributes, S-H-R
7182 0000A015 0F856EF7FFFF        <1>       jnz    loc_permission_denied
7183                              <1>
7184 0000A01B BE[56920100]        <1>         mov     esi, FindFile_Drv
7185 0000A020 BF[9C930100]        <1>         mov     edi, SourceFile_Drv
7186 0000A025 B920000000          <1>       mov    ecx, 32
7187 0000A02A F3A5                <1>       rep    movsd
7188                              <1>
7189                              <1> loc_rename_df_parse_path_name:
7190 0000A02C 8B35[98930100]      <1>       mov    esi, [DestinationFilePath]
7191 0000A032 BF[56920100]        <1>       mov    edi, FindFile_Drv
7192 0000A037 E8E6140000          <1>       call   parse_path_name
7193 0000A03C 7219                <1>       jc     short loc_rename_df_cmd_failed
7194                              <1>
7195                              <1>       ;mov   dh, [RUN_CDRV]
7196 0000A03E 8A35[B6890100]      <1>       mov    dh, [Current_Drv]
7197                              <1>
7198                              <1>       ; 'rename' command is valid only for same dos drive and same dir!
7199                              <1>       ; ('move' command must be used if source file and destination file
7200                              <1>       ; directories are not same!)
7201 0000A044 8A15[56920100]      <1>       mov    dl, [FindFile_Drv]
7202 0000A04A 38F2                <1>       cmp    dl, dh ; are source and destination drives different ?!
7203 0000A04C 7509                <1>       jne    short loc_rename_df_cmd_failed ; yes!
7204                              <1>
7205                              <1> rename_df_check_dirname_exists:
7206 0000A04E 803D[57920100]00    <1>       cmp    byte [FindFile_Directory], 0
7207 0000A055 760B                <1>       jna    short rename_df_check_filename_exists
7208                              <1>
7209                              <1>       ; different source file and destination file directories !
7210                              <1> loc_rename_df_cmd_failed:
7211 0000A057 B801000000          <1>       mov    eax, 1 ; TRDOS 'Bad command or file name' error
7212 0000A05C F9                  <1>       stc
7213 0000A05D E91DF7FFFF          <1>       jmp    loc_file_rw_cmd_failed
7214                              <1>
7215                              <1> rename_df_check_filename_exists:
7216 0000A062 BE[98920100]        <1>       mov    esi, FindFile_Name
7217 0000A067 E8A3F6FFFF          <1>       call   check_filename
7218 0000A06C 0F82BFF7FFFF        <1>       jc     loc_mkdir_invalid_dir_name_chars
7219                              <1>
7220                              <1>       ;mov   [DelFile_FNPointer], esi
7221                              <1>       ;cmp   byte [esi], 20h
7222                              <1>       ;ja    short loc_rename_df_find
7223                              <1>
7224                              <1>       ;mov   dh, [Current_Drv] ; dh has not been changed
7225                              <1>
7226                              <1> rename_df_drv_check_writable:
7227 0000A072 0FB6F6              <1>       movzx  esi, dh
7228                              <1>       ;movzx esi, byte [Current_Drv]
7229 0000A075 81C600010900        <1>       add    esi, Logical_DOSDisks
7230                              <1>
7231 0000A07B 88F2                <1>       mov    dl, dh ; dl = [Current_Drv]
7232 0000A07D 8A7601              <1>       mov    dh, [esi+LD_DiskType]
7233                              <1>
```

```
7234 0000A080 80FE01              <1>       cmp    dh, 1 ; 0 = Invalid
7235 0000A083 7310                <1>       jnb    short rename_df_compare_sf_df_name
7236                              <1>
7237                              <1>       ; 16/10/2016 (13h -> 30)
7238 0000A085 B81E000000          <1>       mov    eax, 30 ; 'Disk write-protected' error
7239 0000A08A 8B1D[98930100]      <1>       mov    ebx, [DestinationFilePath]
7240 0000A090 E9EAF6FFFF          <1>       jmp    loc_file_rw_cmd_failed
7241                              <1>
7242                              <1> rename_df_compare_sf_df_name:
7243 0000A095 BE[98920100]        <1>       mov    esi, FindFile_Name
7244 0000A09A BF[DE930100]        <1>       mov    edi, SourceFile_Name
7245 0000A09F B90C000000          <1>       mov    ecx, 12
7246                              <1> rename_df_compare_sf_df_name_next:
7247 0000A0A4 AC                  <1>       lodsb
7248 0000A0A5 AE                  <1>       scasb
7249 0000A0A6 7506                <1>       jne    short loc_rename_df_find
7250 0000A0A8 08C0                <1>       or     al, al
7251 0000A0AA 74AB                <1>       jz     short loc_rename_df_cmd_failed
7252 0000A0AC E2F6                <1>       loop   rename_df_compare_sf_df_name_next
7253                              <1>
7254                              <1> loc_rename_df_find:
7255                              <1>       ;mov   esi, [DelFile_FNPointer]
7256 0000A0AE BE[98920100]        <1>       mov    esi, FindFile_Name
7257                              <1>
7258 0000A0B3 6631C0              <1>       xor    ax, ax ; Any
7259 0000A0B6 E896F2FFFF          <1>       call   find_first_file
7260                              <1>       ;jnc   short loc_rename_df_found
7261                              <1>       ; 29/12/2017
7262 0000A0BB 0F83C8F6FFFF        <1>       jnc    loc_permission_denied
7263                              <1>
7264                              <1> loc_rename_df_check_error_code:
7265                              <1>       ;cmp   eax, 2
7266 0000A0C1 3C02                <1>       cmp    al, 2 ; Not found error
7267 0000A0C3 7406                <1>       je     short rename_df_move_find_struct_to_dest
7268 0000A0C5 F9                  <1>       stc
7269 0000A0C6 E9B4F6FFFF          <1>       jmp    loc_file_rw_cmd_failed
7270                              <1>
7271                              <1> ;loc_rename_df_found:
7272                              <1>       ; 05/11/2016
7273                              <1>       ; Permission denied error
7274                              <1>       ;mov   eax, ERR_PERM_DENIED ; 29/12/2017
7275                              <1>       ;stc
7276                              <1>       ;jmp   loc_permission_denied  ; 06/11/2016
7277                              <1>
7278                              <1> rename_df_move_find_struct_to_dest:
7279 0000A0CB BE[56920100]        <1>         mov     esi, FindFile_Drv
7280 0000A0D0 BF[1C940100]        <1>         mov     edi, DestinationFile_Drv
7281 0000A0D5 B920000000          <1>       mov    ecx, 32
7282 0000A0DA F3A5                <1>       rep    movsd
7283                              <1>
7284                              <1> loc_rename_df_process_q_sf:
7285                              <1>       ;mov   ecx, 12
7286 0000A0DC B10C                <1>       mov    cl, 12
7287 0000A0DE BE[DE930100]        <1>       mov    esi, SourceFile_Name
7288 0000A0E3 BF[D3440100]        <1>       mov    edi, Rename_OldName
7289                              <1> rename_df_process_q_nml_1_sf:
7290 0000A0E8 AC                  <1>       lodsb
7291 0000A0E9 3C20                <1>         cmp  al, 20h
7292 0000A0EB 7603                <1>         jna  short rename_df_process_q_nml_2_sf
7293 0000A0ED AA                  <1>       stosb
7294 0000A0EE E2F8                <1>       loop   rename_df_process_q_nml_1_sf
7295                              <1>
7296                              <1> rename_df_process_q_nml_2_sf:
7297 0000A0F0 C60700              <1>       mov    byte [edi], 0
7298                              <1>
7299                              <1> loc_rename_df_process_q_df:
7300                              <1>       ;mov   ecx, 12
7301 0000A0F3 B10C                <1>       mov    cl, 12
7302 0000A0F5 BE[5E940100]        <1>       mov    esi, DestinationFile_Name
7303 0000A0FA BF[E4440100]        <1>       mov    edi, Rename_NewName
7304                              <1> rename_df_process_q_nml_1_df:
7305 0000A0FF AC                  <1>       lodsb
7306 0000A100 3C20                <1>       cmp    al, 20h
7307 0000A102 7603                <1>       jna    short loc_rename_df_process_q_nml_2_df
7308 0000A104 AA                  <1>       stosb
7309 0000A105 E2F8                <1>       loop   rename_df_process_q_nml_1_df
7310                              <1>
7311                              <1> loc_rename_df_process_q_nml_2_df:
7312 0000A107 C60700              <1>       mov    byte [edi], 0
7313                              <1>
7314                              <1> loc_rename_confirmation_question:
7315 0000A10A BE[AB440100]        <1>       mov    esi, Msg_DoYouWantRename
7316 0000A10F E811D3FFFF          <1>       call   print_msg
7317                              <1>
7318 0000A114 A0[F9930100]        <1>       mov    al, [SourceFile_DirEntry+11] ; Attributes
7319 0000A119 2410                <1>       and    al, 10h
7320 0000A11B 750C                <1>       jnz    short rename_confirmation_question_dir
7321                              <1>
7322                              <1> rename_confirmation_question_file:
7323 0000A11D BE[C2440100]        <1>       mov    esi, Rename_File
7324 0000A122 E8FED2FFFF          <1>       call   print_msg
7325 0000A127 EB0A                <1>       jmp    short rename_confirmation_question_as
7326                              <1>
7327                              <1> rename_confirmation_question_dir:
7328 0000A129 BE[C8440100]        <1>       mov    esi, Rename_Directory
7329 0000A12E E8F2D2FFFF          <1>       call   print_msg
7330                              <1>
7331                              <1> rename_confirmation_question_as:
7332 0000A133 BE[D3440100]        <1>       mov    esi, Rename_OldName
7333 0000A138 E8E8D2FFFF          <1>       call   print_msg
7334 0000A13D BE[E0440100]        <1>       mov    esi, Msg_File_rename_as
7335 0000A142 E8DED2FFFF          <1>       call   print_msg
7336 0000A147 BE[07440100]        <1>       mov    esi, Msg_YesNo
7337 0000A14C E8D4D2FFFF          <1>       call   print_msg
7338                              <1>
```

```
7339                                    <1> loc_rename_ask_again:
7340 0000A151 30E4                      <1>         xor     ah, ah
7341 0000A153 E88B6DFFFF                <1>         call    int16h
7342 0000A158 3C1B                      <1>         cmp     al, 1Bh
7343 0000A15A 740F                      <1>         je      short loc_do_not_rename_file
7344 0000A15C 24DF                      <1>         and     al, 0DFh
7345 0000A15E A2[11440100]              <1>         mov     [Y_N_nextline], al
7346 0000A163 3C59                      <1>         cmp     al, 'Y'
7347 0000A165 7404                      <1>         je      short loc_yes_rename_file
7348 0000A167 3C4E                      <1>         cmp     al, 'N'
7349 0000A169 75E6                      <1>         jne     short loc_rename_ask_again
7350                                    <1>
7351                                    <1> loc_do_not_rename_file:
7352                                    <1> loc_yes_rename_file:
7353 0000A16B E848F7FFFF                <1>         call    y_n_answer ; 29/12/2017
7354                                    <1>         ;cmp    al, 'Y' ; 'yes'
7355                                    <1>         ;cmc
7356                                    <1>           ;jnc  loc_file_rw_restore_retn
7357 0000A170 3C4E                      <1>         cmp     al, 'N' ; 'no'
7358 0000A172 0F8407F6FFFF              <1>           je    loc_file_rw_restore_retn
7359                                    <1>
7360 0000A178 BE[E4440100]              <1>         mov     esi, Rename_NewName
7361 0000A17D 668B0D[16940100]          <1>         mov     cx, [SourceFile_DirEntryNumber]
7362 0000A184 66A1[02940100]            <1>         mov     ax, [SourceFile_DirEntry+20] ; First Cluster, HW
7363 0000A18A C1E010                    <1>         shl     eax, 16 ; 13/11/2017
7364 0000A18D 66A1[08940100]            <1>         mov     ax, [SourceFile_DirEntry+26] ; First Cluster, LW
7365                                    <1>
7366 0000A193 0FB61D[EB930100]          <1>         movzx   ebx, byte [SourceFile_LongNameEntryLength]
7367 0000A19A E8CB1B0000                <1>         call    rename_directory_entry
7368 0000A19F E9FBF6FFFF                <1>         jmp     loc_rename_file_ok
7369                                    <1> ;loc_rename_file_ok:
7370                                    <1> ;       jc      loc_run_cmd_failed
7371                                    <1> ;       mov     esi, Msg_OK
7372                                    <1> ;       call    proc_printmsg
7373                                    <1> ;       jmp     loc_file_rw_restore_retn
7374                                    <1>
7375                                    <1> move_file:
7376                                    <1>         ; 11/03/2016
7377                                    <1>         ; 09/03/2016
7378                                    <1>         ; 08/03/2016 (TRDOS 386 = TRDOS v2.0)
7379                                    <1>         ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_move')
7380                                    <1>         ; 23/04/2011
7381                                    <1>
7382                                    <1> get_move_source_fchar:
7383                                    <1>         ; esi = file name
7384 0000A1A4 803E20                    <1>         cmp     byte [esi], 20h
7385 0000A1A7 7614                      <1>           jna   short loc_move_nofilename_retn
7386                                    <1>
7387 0000A1A9 8935[94930100]            <1>         mov     [SourceFilePath], esi
7388                                    <1>
7389                                    <1> move_scan_source_file:
7390 0000A1AF 46                        <1>         inc     esi
7391 0000A1B0 803E20                    <1>         cmp     byte [esi], 20h
7392 0000A1B3 7409                      <1>           je      short move_scan_destination_1
7393                                    <1>         ;jb     short loc_move_nofilename_retn
7394 0000A1B5 0F8286ECFFFF              <1>         jb      loc_cmd_failed
7395 0000A1BB EBF2                      <1>         jmp     short move_scan_source_file
7396                                    <1>
7397                                    <1> loc_move_nofilename_retn:
7398 0000A1BD C3                        <1>         retn
7399                                    <1>
7400                                    <1> move_scan_destination_1:
7401 0000A1BE C60600                    <1>         mov     byte [esi], 0
7402                                    <1>
7403                                    <1> move_scan_destination_2:
7404 0000A1C1 46                        <1>         inc     esi
7405 0000A1C2 803E20                    <1>         cmp     byte [esi], 20h
7406 0000A1C5 74FA                      <1>         je      short move_scan_destination_2
7407                                    <1>         ;jb     short loc_move_nofilename_retn
7408 0000A1C7 0F8274ECFFFF              <1>         jb      loc_cmd_failed
7409                                    <1>
7410 0000A1CD 8935[98930100]            <1>         mov     [DestinationFilePath], esi
7411                                    <1>
7412                                    <1> move_scan_destination_3:
7413 0000A1D3 46                        <1>         inc     esi
7414 0000A1D4 803E20                    <1>         cmp     byte [esi], 20h
7415 0000A1D7 77FA                      <1>         ja      short move_scan_destination_3
7416 0000A1D9 C60600                    <1>         mov     byte [esi], 0
7417                                    <1>
7418                                    <1> loc_move_scan_destination_OK:
7419 0000A1DC 8B35[94930100]            <1>         mov     esi, [SourceFilePath]
7420 0000A1E2 8B3D[98930100]            <1>         mov     edi, [DestinationFilePath]
7421                                    <1>
7422 0000A1E8 B001                      <1>         mov     al, 1  ; move procedure Phase 1
7423 0000A1EA E8F71B0000                <1>         call    move_source_file_to_destination_file
7424 0000A1EF 7328                      <1>         jnc     short move_source_file_to_destination_question
7425                                    <1>
7426                                    <1> loc_move_cmd_failed_1:
7427 0000A1F1 08C0                      <1>         or      al, al
7428 0000A1F3 0F8448ECFFFF              <1>         jz      loc_cmd_failed
7429 0000A1F9 3C11                      <1>         cmp     al, 11h
7430 0000A1FB 740D                      <1>         je      short loc_msg_not_same_device
7431                                    <1>         ;cmp    al, 05h
7432                                    <1>         ;cmp    al, ERR_PERM_DENIED ; 29/12/2017
7433                                    <1>         ;jne    loc_run_cmd_failed
7434                                    <1>         ;jmp    loc_permission_denied
7435 0000A1FD 3C0B                      <1>         cmp     al, ERR_PERM_DENIED
7436 0000A1FF 0F8484F5FFFF              <1>         je      loc_permission_denied
7437 0000A205 E962ECFFFF                <1>         jmp     loc_run_cmd_failed
7438                                    <1>
7439                                    <1>         ;mov    esi, Msg_Permission_denied
7440                                    <1>         ;call   print_msg
7441                                    <1>         ;jmp    loc_file_rw_restore_retn
7442                                    <1>
7443                                    <1> loc_msg_not_same_device:
```

```
7444 0000A20A BE[F1440100]        <1>       mov    esi, msg_not_same_drv
7445 0000A20F E811D2FFFF          <1>       call   print_msg
7446 0000A214 E966F5FFFF          <1>       jmp    loc_file_rw_restore_retn
7447                              <1>
7448                              <1> move_source_file_to_destination_question:
7449 0000A219 A0[9C930100]        <1>         mov     al, [SourceFile_Drv]
7450 0000A21E 0441                <1>       add    al, 'A'
7451 0000A220 A2[53450100]        <1>       mov    [msg_source_file_drv], al
7452 0000A225 A0[1C940100]        <1>         mov     al, [DestinationFile_Drv]
7453 0000A22A 0441                <1>       add    al, 'A'
7454 0000A22C A2[72450100]        <1>       mov    [msg_destination_file_drv], al
7455                              <1>
7456 0000A231 57                  <1>       push   edi ; *
7457                              <1>
7458 0000A232 BE[37450100]        <1>       mov    esi, msg_source_file
7459 0000A237 E8E9D1FFFF          <1>       call   print_msg
7460 0000A23C BE[9D930100]        <1>       mov    esi, SourceFile_Directory
7461 0000A241 803E20              <1>       cmp    byte [esi], 20h
7462 0000A244 7605                <1>       jna    short msftdfq_sfn
7463 0000A246 E8DAD1FFFF          <1>       call   print_msg
7464                              <1> msftdfq_sfn:
7465 0000A24B BE[DE930100]        <1>       mov    esi, SourceFile_Name
7466 0000A250 E8D0D1FFFF          <1>       call   print_msg
7467 0000A255 BE[56450100]        <1>       mov    esi, msg_destination_file
7468 0000A25A E8C6D1FFFF          <1>       call   print_msg
7469 0000A25F BE[1D940100]        <1>       mov    esi, DestinationFile_Directory
7470 0000A264 803E20              <1>       cmp    byte [esi], 20h
7471 0000A267 7605                <1>       jna    short msftdfq_dfn
7472 0000A269 E8B7D1FFFF          <1>       call   print_msg
7473                              <1> msftdfq_dfn:
7474 0000A26E BE[5E940100]        <1>       mov    esi, DestinationFile_Name
7475 0000A273 E8ADD1FFFF          <1>       call   print_msg
7476 0000A278 BE[75450100]        <1>       mov    esi, msg_copy_nextline
7477 0000A27D E8A3D1FFFF          <1>       call   print_msg
7478 0000A282 BE[75450100]        <1>       mov    esi, msg_copy_nextline
7479 0000A287 E899D1FFFF          <1>       call   print_msg
7480                              <1>
7481                              <1> loc_move_ask_for_new_file_yes_no:
7482 0000A28C BE[03450100]        <1>       mov    esi, Msg_DoYouWantMoveFile
7483 0000A291 E88FD1FFFF          <1>       call   print_msg
7484 0000A296 BE[07440100]        <1>       mov    esi, Msg_YesNo
7485 0000A29B E885D1FFFF          <1>       call   print_msg
7486                              <1> loc_move_ask_for_new_file_again:
7487 0000A2A0 30E4                <1>       xor    ah, ah
7488 0000A2A2 E83C6CFFFF          <1>       call   int16h
7489 0000A2A7 3C1B                <1>       cmp    al, 1Bh
7490                              <1>       ;je    short loc_do_not_move_file
7491 0000A2A9 7441                <1>       je     short loc_move_y_n_escape
7492 0000A2AB 24DF                <1>       and    al, 0DFh
7493 0000A2AD A2[11440100]        <1>         mov     [Y_N_nextline], al
7494 0000A2B2 3C59                <1>       cmp    al, 'Y'
7495 0000A2B4 7404                <1>       je     short loc_yes_move_file
7496 0000A2B6 3C4E                <1>       cmp    al, 'N'
7497 0000A2B8 75E6                <1>       jne    short loc_move_ask_for_new_file_again
7498                              <1>
7499                              <1> loc_do_not_move_file:
7500                              <1> loc_yes_move_file:
7501 0000A2BA E8F9F5FFFF          <1>       call   y_n_answer ; 29/12/2017
7502 0000A2BF 5F                  <1>       pop    edi ; *
7503                              <1>       ;cmp   al, 'Y' ; 'yes'
7504                              <1>       ;cmc
7505                              <1>         ;jnc loc_file_rw_restore_retn
7506 0000A2C0 3C4E                <1>       cmp    al, 'N' ; 'no'
7507 0000A2C2 0F84B7F4FFFF        <1>         je   loc_file_rw_restore_retn
7508                              <1>
7509                              <1> loc_move_yes_move_file:
7510 0000A2C8 B002                <1>       mov    al, 2 ; move procedure Phase 2
7511 0000A2CA E8171B0000          <1>       call   move_source_file_to_destination_file
7512                              <1>       ;jc    short loc_move_cmd_failed_2
7513 0000A2CF 0F83D0F5FFFF        <1>         jnc     move_source_file_to_destination_OK
7514                              <1>
7515                              <1> ;move_source_file_to_destination_OK:
7516                              <1> ;     mov    esi, Msg_OK
7517                              <1> ;     call   print_msg
7518                              <1> ;     jmp    loc_file_rw_restore_retn
7519                              <1>
7520                              <1> loc_move_cmd_failed_2:
7521 0000A2D5 3C27                <1>       cmp    al, 27h
7522 0000A2D7 0F858FEBFFFF        <1>       jne    loc_run_cmd_failed
7523                              <1>
7524 0000A2DD BE[1C450100]        <1>       mov    esi, msg_insufficient_disk_space
7525 0000A2E2 E83ED1FFFF          <1>       call   print_msg
7526                              <1>
7527 0000A2E7 E993F4FFFF          <1>       jmp    loc_file_rw_restore_retn
7528                              <1>
7529                              <1> loc_move_y_n_escape:
7530 0000A2EC B04E                <1>       mov    al, 'N' ; 'no'
7531 0000A2EE EBCA                <1>       jmp    short loc_do_not_move_file
7532                              <1>
7533                              <1> copy_file:
7534                              <1>       ; 15/10/2016
7535                              <1>       ; 24/03/2016
7536                              <1>       ; 21/03/2016
7537                              <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
7538                              <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_copy')
7539                              <1>       ; 01/08/2010
7540                              <1>
7541                              <1> get_copy_source_fchar:
7542                              <1>       ; esi = file name
7543 0000A2F0 803E20              <1>       cmp    byte [esi], 20h
7544 0000A2F3 7614                <1>         jna     short loc_copy_nofilename_retn
7545                              <1>
7546 0000A2F5 8935[94930100]      <1>       mov    [SourceFilePath], esi
7547                              <1>
7548                              <1> copy_scan_source_file:
```

```
7549 0000A2FB 46                <1>       inc    esi
7550 0000A2FC 803E20            <1>       cmp    byte [esi], 20h
7551 0000A2FF 7409              <1>       je     short copy_scan_destination_1
7552                            <1>       ;jb    short loc_copy_nofilename_retn
7553 0000A301 0F823AEBFFFF      <1>       jb     loc_cmd_failed
7554 0000A307 EBF2              <1>       jmp    short copy_scan_source_file
7555                            <1>
7556                            <1> loc_copy_nofilename_retn:
7557 0000A309 C3                <1>       retn
7558                            <1>
7559                            <1> copy_scan_destination_1:
7560 0000A30A C60600            <1>       mov    byte [esi], 0
7561                            <1>
7562                            <1> copy_scan_destination_2:
7563 0000A30D 46                <1>       inc    esi
7564 0000A30E 803E20            <1>       cmp    byte [esi], 20h
7565 0000A311 74FA              <1>       je     short copy_scan_destination_2
7566                            <1>       ;jb    short loc_copy_nofilename_retn
7567 0000A313 0F8228EBFFFF      <1>       jb     loc_cmd_failed
7568                            <1>
7569 0000A319 8935[98930100]    <1>       mov    [DestinationFilePath], esi
7570                            <1>
7571                            <1> copy_scan_destination_3:
7572 0000A31F 46                <1>       inc    esi
7573 0000A320 803E20            <1>       cmp    byte [esi], 20h
7574 0000A323 77FA              <1>       ja     short copy_scan_destination_3
7575 0000A325 C60600            <1>       mov    byte [esi], 0
7576                            <1>
7577                            <1> loc_copy_save_current_drive:
7578 0000A328 8A35[B6890100]    <1>       mov    dh, [Current_Drv]
7579 0000A32E 8835[12910100]    <1>       mov    [RUN_CDRV], dh
7580                            <1>
7581                            <1> copy_source_file_to_destination_phase_1:
7582 0000A334 8B35[94930100]    <1>       mov    esi, [SourceFilePath]
7583 0000A33A 8B3D[98930100]    <1>       mov    edi, [DestinationFilePath]
7584                            <1>
7585 0000A340 B001              <1>       mov    al, 1  ; copy procedure Phase 1
7586 0000A342 E83C1D0000        <1>       call   copy_source_file_to_destination_file
7587 0000A347 732B              <1>       jnc    short copy_source_file_to_destination_question
7588                            <1>
7589                            <1> loc_copy_cmd_failed_1:
7590                            <1>       ; 18/03/2016 (restore current drive and directory)
7591 0000A349 08C0              <1>       or     al, al
7592 0000A34B 7507              <1>       jnz    short loc_copy_cmd_failed_2
7593                            <1>
7594 0000A34D FEC0              <1>         inc     al ; mov al, 1 ; Bad command or file name !
7595 0000A34F E918EBFFFF        <1>       jmp    loc_run_cmd_failed
7596                            <1>
7597                            <1> loc_copy_cmd_failed_2:
7598 0000A354 3C27              <1>       cmp    al, 27h ; Insufficient disk space
7599 0000A356 740D              <1>       je     short loc_file_write_insuff_disk_space_msg
7600                            <1>
7601                            <1>       ; 29/12/2017
7602                            <1>       ;cmp   al, 05h
7603 0000A358 3C0B              <1>       cmp    al, ERR_PERM_DENIED
7604 0000A35A 0F850CEBFFFF      <1>       jne    loc_run_cmd_failed
7605                            <1>
7606 0000A360 E924F4FFFF        <1>       jmp    loc_permission_denied
7607                            <1>
7608                            <1> loc_file_write_insuff_disk_space_msg:
7609 0000A365 BE[1C450100]      <1>       mov    esi, msg_insufficient_disk_space
7610 0000A36A E8B6D0FFFF        <1>       call   print_msg
7611 0000A36F E90BF4FFFF        <1>         jmp     loc_file_rw_restore_retn
7612                            <1>
7613                            <1> copy_source_file_to_destination_question:
7614 0000A374 57                <1>       push   edi ; *
7615                            <1>
7616                            <1>       ; dh = source file attributes
7617                            <1>       ; dl > 0 -> destination file found
7618 0000A375 20D2              <1>       and    dl, dl
7619 0000A377 7449              <1>       jz     short copy_source_file_to_destination_pass_owrq
7620                            <1>
7621                            <1> loc_copy_ask_for_owr_yes_no:
7622 0000A379 BE[78450100]      <1>       mov    esi, Msg_DoYouWantOverWriteFile
7623 0000A37E E8A2D0FFFF        <1>       call   print_msg
7624 0000A383 BE[5E940100]      <1>       mov    esi, DestinationFile_Name
7625 0000A388 E898D0FFFF        <1>       call   print_msg
7626 0000A38D BE[07440100]      <1>       mov    esi, Msg_YesNo
7627 0000A392 E88ED0FFFF        <1>       call   print_msg
7628                            <1>
7629                            <1> loc_copy_ask_for_owr_again:
7630 0000A397 30E4              <1>       xor    ah, ah
7631 0000A399 E8456BFFFF        <1>       call   int16h
7632 0000A39E 3C1B              <1>       cmp    al, 1Bh
7633                            <1>         ;je     loc_do_not_copy_file
7634 0000A3A0 7419              <1>         je      short loc_copy_y_n_escape
7635 0000A3A2 24DF              <1>       and    al, 0DFh
7636 0000A3A4 A2[11440100]      <1>         mov     [Y_N_nextline], al
7637 0000A3A9 3C59              <1>       cmp    al, 'Y'
7638 0000A3AB 0F84B1000000      <1>         je      loc_yes_copy_file
7639 0000A3B1 3C4E              <1>       cmp    al, 'N'
7640 0000A3B3 0F84A9000000      <1>         je      loc_do_not_copy_file
7641 0000A3B9 EBDC              <1>       jmp    short loc_copy_ask_for_owr_again
7642                            <1>
7643                            <1> loc_copy_y_n_escape:
7644 0000A3BB B04E              <1>       mov    al, 'N' ; 'no'
7645 0000A3BD E9A0000000        <1>         jmp     loc_do_not_copy_file
7646                            <1>
7647                            <1> copy_source_file_to_destination_pass_owrq:
7648 0000A3C2 A0[9C930100]      <1>       mov     al, [SourceFile_Drv]
7649 0000A3C7 0441              <1>       add    al, 'A'
7650 0000A3C9 A2[53450100]      <1>       mov    [msg_source_file_drv], al
7651 0000A3CE A0[1C940100]      <1>         mov     al, [DestinationFile_Drv]
7652 0000A3D3 0441              <1>       add    al, 'A'
7653 0000A3D5 A2[72450100]      <1>       mov    [msg_destination_file_drv], al
```

```
7654                                    <1>
7655 0000A3DA BE[37450100]              <1>       mov    esi, msg_source_file
7656 0000A3DF E841D0FFFF                <1>       call   print_msg
7657 0000A3E4 BE[9D930100]              <1>       mov    esi, SourceFile_Directory
7658 0000A3E9 803E20                    <1>       cmp    byte [esi], 20h
7659 0000A3EC 7605                      <1>       jna    short csftdfq_sfn
7660 0000A3EE E832D0FFFF                <1>       call   print_msg
7661                                    <1> csftdfq_sfn:
7662 0000A3F3 BE[DE930100]              <1>       mov    esi, SourceFile_Name
7663 0000A3F8 E828D0FFFF                <1>       call   print_msg
7664 0000A3FD BE[56450100]              <1>       mov    esi, msg_destination_file
7665 0000A402 E81ED0FFFF                <1>       call   print_msg
7666 0000A407 BE[1D940100]              <1>       mov    esi, DestinationFile_Directory
7667 0000A40C 803E20                    <1>       cmp    byte [esi], 20h
7668 0000A40F 7605                      <1>       jna    short csftdfq_dfn
7669 0000A411 E80FD0FFFF                <1>       call   print_msg
7670                                    <1> csftdfq_dfn:
7671 0000A416 BE[5E940100]              <1>       mov    esi, DestinationFile_Name
7672 0000A41B E805D0FFFF                <1>       call   print_msg
7673 0000A420 BE[75450100]              <1>       mov    esi, msg_copy_nextline
7674 0000A425 E8FBCFFFFF                <1>       call   print_msg
7675 0000A42A BE[75450100]              <1>       mov    esi, msg_copy_nextline
7676 0000A42F E8F1CFFFFF                <1>       call   print_msg
7677                                    <1>
7678                                    <1> loc_copy_ask_for_new_file_yes_no:
7679 0000A434 BE[97450100]              <1>       mov    esi, Msg_DoYouWantCopyFile
7680 0000A439 E8E7CFFFFF                <1>       call   print_msg
7681 0000A43E BE[07440100]              <1>       mov    esi, Msg_YesNo
7682 0000A443 E8DDCFFFFF                <1>       call   print_msg
7683                                    <1>
7684                                    <1> loc_copy_ask_for_new_file_again:
7685 0000A448 30E4                      <1>       xor    ah, ah
7686 0000A44A E8946AFFFF                <1>       call   int16h
7687 0000A44F 3C1B                      <1>       cmp    al, 1Bh
7688 0000A451 740F                      <1>       je     short loc_do_not_copy_file
7689 0000A453 24DF                      <1>       and    al, 0DFh
7690 0000A455 A2[11440100]              <1>         mov     [Y_N_nextline], al
7691 0000A45A 3C59                      <1>       cmp    al, 'Y'
7692 0000A45C 7404                      <1>       je     short loc_yes_copy_file
7693 0000A45E 3C4E                      <1>       cmp    al, 'N'
7694 0000A460 75E6                      <1>       jne    short loc_copy_ask_for_new_file_again
7695                                    <1>
7696                                    <1> loc_do_not_copy_file:
7697                                    <1> loc_yes_copy_file:
7698 0000A462 E851F4FFFF                <1>       call   y_n_answer ; 29/12/2017
7699 0000A467 5F                        <1>       pop    edi ; *
7700                                    <1>       ;cmp   al, 'Y' ; 'yes'
7701                                    <1>       ;cmc
7702                                    <1>         ;jnc loc_file_rw_restore_retn
7703 0000A468 3C4E                      <1>       cmp    al, 'N' ; 'no'
7704 0000A46A 0F840FF3FFFF              <1>         je   loc_file_rw_restore_retn
7705                                    <1>
7706                                    <1> copy_source_file_to_destination_pass_q:
7707 0000A470 B002                      <1>       mov    al, 2  ; copy procedure Phase 2
7708 0000A472 E80C1C0000                <1>       call   copy_source_file_to_destination_file
7709                                    <1>       ;jc    short loc_file_write_check_disk_space_err
7710                                    <1>
7711                                    <1>       ; 24/03/2016
7712                                    <1>       ;push  cx
7713 0000A477 51                        <1>       push   ecx ; 29/12/2017
7714 0000A478 BE[75450100]              <1>       mov    esi, msg_copy_nextline
7715 0000A47D E8A3CFFFFF                <1>       call   print_msg
7716 0000A482 58                        <1>       pop    eax ; 29/12/2017
7717                                    <1>       ;;pop  cx
7718                                    <1>       ;pop   ax
7719                                    <1>
7720                                    <1>       ;or    cl, cl
7721 0000A483 08C0                      <1>       or     al, al
7722 0000A485 7419                      <1>       jz     short copy_source_file_to_destination_OK
7723                                    <1>
7724                                    <1>       ; 15/10/2016 (1Dh -> 18)
7725                                    <1>       ; 18/03/2016 (1Dh)
7726                                    <1>       ;cmp   cl, 18 ; write error
7727 0000A487 3C12                      <1>       cmp    al, 18
7728 0000A489 7506                      <1>       jne    short copy_source_file_to_destination_not_OK
7729                                    <1>       ;
7730                                    <1>       ;mov   al, cl ; error number (write fault!)
7731 0000A48B F9                        <1>       stc
7732 0000A48C E9EEF2FFFF                <1>       jmp    loc_file_rw_cmd_failed
7733                                    <1>
7734                                    <1> copy_source_file_to_destination_not_OK:
7735 0000A491 BE[B0450100]              <1>       mov    esi, Msg_read_file_error_before_EOF
7736 0000A496 E88ACFFFFF                <1>       call   print_msg
7737 0000A49B E9DFF2FFFF                <1>       jmp    loc_file_rw_restore_retn
7738                                    <1>
7739                                    <1> copy_source_file_to_destination_OK:
7740 0000A4A0 BE[15440100]              <1>       mov    esi, Msg_OK
7741 0000A4A5 E87BCFFFFF                <1>       call   print_msg
7742                                    <1>
7743 0000A4AA E9D0F2FFFF                <1>       jmp    loc_file_rw_restore_retn
7744                                    <1>
7745                                    <1> ;loc_file_write_check_disk_space_err:
7746                                    <1>       ;cmp   al, 27h ; Insufficient disk space
7747                                    <1>       ;je    loc_file_write_insuff_disk_space_msg
7748                                    <1>         ;jb  loc_file_rw_cmd_failed
7749                                    <1>
7750                                    <1>       ;call  print_misc_error_msg ; 15/03/2016
7751                                    <1>         ;jmp loc_file_rw_restore_retn
7752                                    <1>
7753                                    <1> change_fs_file_attributes:
7754                                    <1>       ; 04/03/2016 ; Temporary
7755                                    <1>       ; AL = File or directory attributes
7756                                    <1>       ; AH = 0 -> Attributes are in MS-DOS format
7757                                    <1>       ; AH > 0 -> Attributes are in SINGLIX format
7758                                    <1>       ;push  ebx
```

```
7759                            <1>       ; ... do somethings here ...
7760                            <1>       ;pop   ebx
7761                            <1>       ; BL = File or directory attributes
7762 0000A4AF C3                <1>       retn
7763                            <1>
7764                            <1> set_get_env:
7765                            <1>       ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
7766                            <1>       ; 02/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_set')
7767                            <1>       ; 2005 - 28/08/2011
7768                            <1> get_setenv_fchar:
7769                            <1>       ; esi = environment variable/string
7770 0000A4B0 8A06              <1>       mov    al, [esi]
7771 0000A4B2 3C20              <1>       cmp    al, 20h
7772 0000A4B4 771E              <1>       ja     short loc_find_env
7773                            <1>
7774 0000A4B6 BE00300900        <1>       mov    esi, Env_Page
7775                            <1> loc_print_setline:
7776 0000A4BB 803E00            <1>       cmp    byte [esi], 0
7777 0000A4BE 7613              <1>       jna    short loc_setenv_retn
7778 0000A4C0 E860CFFFFF        <1>       call   print_msg
7779 0000A4C5 56                <1>       push   esi
7780 0000A4C6 BE[1F4C0100]      <1>       mov    esi, nextline
7781 0000A4CB E855CFFFFF        <1>       call   print_msg
7782 0000A4D0 5E                <1>       pop    esi
7783 0000A4D1 EBE8              <1>       jmp    short loc_print_setline
7784                            <1>
7785                            <1> loc_setenv_retn:
7786 0000A4D3 C3                <1>       retn
7787                            <1>
7788                            <1> loc_find_env:
7789 0000A4D4 3C3D              <1>       cmp    al, '='
7790 0000A4D6 0F8465E9FFFF      <1>       je     loc_cmd_failed
7791                            <1>
7792 0000A4DC 56                <1>       push   esi
7793                            <1> loc_repeat_env_equal_check:
7794 0000A4DD 46                <1>       inc    esi
7795 0000A4DE 803E3D            <1>       cmp    byte [esi], '='
7796 0000A4E1 7431              <1>       je     short pass_env_equal_check
7797 0000A4E3 803E20            <1>       cmp    byte [esi], 20h
7798 0000A4E6 73F5              <1>       jnb    short loc_repeat_env_equal_check
7799 0000A4E8 C60600            <1>       mov    byte [esi], 0
7800 0000A4EB 5E                <1>       pop    esi
7801 0000A4EC BF[B68A0100]      <1>       mov    edi, TextBuffer ; out buffer
7802 0000A4F1 B9FF000000        <1>       mov    ecx, 255 ; maximum size (limit)
7803 0000A4F6 30C0              <1>       xor    al, al ; 0 -> use [ESI]
7804 0000A4F8 E89E000000        <1>       call   get_environment_string
7805 0000A4FD 72D4              <1>       jc     short loc_setenv_retn
7806                            <1>
7807 0000A4FF BE[B68A0100]      <1>       mov    esi, TextBuffer
7808 0000A504 E81CCFFFFF        <1>       call   print_msg
7809 0000A509 BE[1F4C0100]      <1>       mov    esi, nextline
7810 0000A50E E812CFFFFF        <1>       call   print_msg
7811                            <1>
7812 0000A513 C3                <1>       retn
7813                            <1>
7814                            <1> pass_env_equal_check:
7815 0000A514 46                <1>       inc    esi
7816 0000A515 803E20            <1>       cmp    byte [esi], 20h
7817 0000A518 73FA              <1>       jnb    short pass_env_equal_check
7818 0000A51A C60600            <1>       mov    byte [esi], 0
7819                            <1>
7820                            <1> loc_call_set_env_string:
7821 0000A51D 5E                <1>       pop    esi
7822 0000A51E E83B010000        <1>       call   set_environment_string
7823 0000A523 73AE              <1>       jnc    short loc_setenv_retn
7824                            <1>
7825                            <1> loc_set_cmd_failed:
7826 0000A525 3C08              <1>       cmp    al, 08h
7827 0000A527 0F8514E9FFFF      <1>       jne    loc_cmd_failed
7828                            <1>
7829 0000A52D BE[F0450100]      <1>       mov    esi, Msg_No_Set_Space
7830 0000A532 E8EECEFFFF        <1>       call   print_msg
7831                            <1>
7832 0000A537 C3                <1>       retn
7833                            <1>
7834                            <1> set_get_path:
7835                            <1>       ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
7836                            <1>       ; 03/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_path')
7837                            <1>       ; 2005
7838                            <1> get_path_fchar:
7839                            <1>       ; esi = path
7840 0000A538 803E20            <1>       cmp    byte [esi], 20h
7841 0000A53B 7737              <1>       ja     short loc_set_path
7842                            <1>
7843 0000A53D BE00300900        <1>       mov    esi, Env_Page
7844                            <1> loc_print_path:
7845 0000A542 803E00            <1>       cmp    byte [esi], 0
7846 0000A545 762C              <1>       jna    short loc_path_retn
7847                            <1>
7848 0000A547 BE[4F400100]      <1>       mov    esi, Cmd_Path ; 'PATH' address
7849 0000A54C BF[B68A0100]      <1>       mov    edi, TextBuffer ; out buffer
7850 0000A551 30C0              <1>       xor    al, al  ; use [ESI]
7851 0000A553 B9FF000000        <1>       mov    ecx, 255 ; maximum size (limit)
7852 0000A558 E83E000000        <1>       call   get_environment_string
7853 0000A55D 7214              <1>       jc     short loc_path_retn
7854                            <1>
7855 0000A55F BE[B68A0100]      <1>       mov    esi, TextBuffer
7856 0000A564 E8BCCEFFFF        <1>       call   print_msg
7857 0000A569 BE[1F4C0100]      <1>       mov    esi, nextline
7858 0000A56E E8B2CEFFFF        <1>       call   print_msg
7859                            <1>
7860                            <1> loc_path_retn:
7861 0000A573 C3                <1>       retn
7862                            <1>
7863                            <1> loc_set_path:
```

```
7864 0000A574 56                <1>       push   esi
7865                            <1> loc_set_path_find_end:
7866 0000A575 46                <1>       inc    esi
7867 0000A576 803E20            <1>       cmp    byte [esi], 20h
7868 0000A579 73FA              <1>       jnb    short loc_set_path_find_end
7869 0000A57B C60600            <1>       mov    byte [esi], 0
7870                            <1> loc_set_path_header:
7871 0000A57E 5E                <1>       pop    esi
7872                            <1> set_path_x: ; 31/12/2017 ('syspath')
7873 0000A57F 4E                <1>       dec    esi
7874 0000A580 C6063D            <1>       mov    byte [esi], '='
7875 0000A583 4E                <1>       dec    esi
7876 0000A584 C60648            <1>       mov    byte [esi], 'H'
7877 0000A587 4E                <1>       dec    esi
7878 0000A588 C60654            <1>       mov    byte [esi], 'T'
7879 0000A58B 4E                <1>       dec    esi
7880 0000A58C C60641            <1>       mov    byte [esi], 'A'
7881 0000A58F 4E                <1>       dec    esi
7882 0000A590 C60650            <1>       mov    byte [esi], 'P'
7883                            <1>
7884                            <1> loc_path_call_set_env_string:
7885 0000A593 E8C6000000        <1>       call   set_environment_string
7886 0000A598 728B              <1>         jc   short loc_set_cmd_failed
7887                            <1>
7888 0000A59A C3                <1>       retn
7889                            <1>
7890                            <1> get_environment_string:
7891                            <1>       ; 12/04/2016
7892                            <1>       ; 11/04/2016
7893                            <1>       ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
7894                            <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
7895                            <1>       ; 28/08/2011
7896                            <1>       ; INPUT->
7897                            <1>       ;      EDI = Output buffer
7898                            <1>       ;      CX = Buffer length (<= ENV_PAGE_SIZE)
7899                            <1>       ;
7900                            <1>       ;      AL > 0 = AL = String sequence number
7901                            <1>       ;      AL = 0 -> ESI = ASCIIZ Set word
7902                            <1>       ;             (environment variable)
7903                            <1>       ; OUTPUT ->
7904                            <1>       ;      ESI is not changed
7905                            <1>       ;      EDI is not changed
7906                            <1>       ;      EAX = String length (with zero tail)
7907                            <1>       ;      EDX = Environment variables page address
7908                            <1>       ;      CF = 1 -> Not found (EAX not valid)
7909                            <1>       ;
7910                            <1>       ; (Modified registers: EAX, EDX)
7911                            <1>
7912 0000A59B BA00300900        <1>       mov    edx, Env_Page
7913 0000A5A0 803A00            <1>       cmp    byte [edx], 0
7914 0000A5A3 7474              <1>       jz     short get_env_string_with_word_stc_retn
7915                            <1>
7916 0000A5A5 66890D[20950100]  <1>       mov    [env_var_length], cx
7917                            <1>
7918 0000A5AC 51                <1>       push   ecx ; *
7919 0000A5AD 56                <1>       push   esi ; **
7920                            <1>
7921 0000A5AE 08C0              <1>       or     al, al
7922 0000A5B0 7449              <1>       jz     short get_env_string_with_word
7923                            <1>
7924                            <1> get_env_string_with_seq_number:
7925 0000A5B2 B101              <1>       mov    cl, 1
7926 0000A5B4 88C5              <1>       mov    ch, al
7927 0000A5B6 31C0              <1>       xor    eax, eax
7928 0000A5B8 89D6              <1>       mov    esi, edx ; Env_Page
7929                            <1>
7930                            <1> get_env_string_seq_number_check:
7931 0000A5BA 38CD              <1>       cmp    ch, cl
7932 0000A5BC 7726              <1>       ja     short get_env_string_seq_number_next
7933                            <1>
7934                            <1> get_env_string_move_to_buff:
7935 0000A5BE 57                <1>       push   edi ; ***
7936                            <1>
7937 0000A5BF 29D2              <1>       sub    edx, edx
7938                            <1>
7939                            <1> get_env_string_seq_number_repeat1:
7940 0000A5C1 42                <1>       inc    edx
7941 0000A5C2 AC                <1>       lodsb
7942 0000A5C3 AA                <1>       stosb
7943                            <1>
7944 0000A5C4 66FF0D[20950100]  <1>       dec    word [env_var_length]
7945 0000A5CB 7508              <1>       jnz    short get_env_string_seq_number_repeat3
7946                            <1>
7947                            <1> get_env_string_seq_number_repeat2:
7948 0000A5CD 20C0              <1>       and    al, al
7949 0000A5CF 7408              <1>       jz     short get_env_string_seq_number_ok
7950 0000A5D1 42                <1>       inc    edx
7951 0000A5D2 AC                <1>       lodsb
7952 0000A5D3 EBF8              <1>       jmp    short get_env_string_seq_number_repeat2
7953                            <1>
7954                            <1> get_env_string_seq_number_repeat3:
7955 0000A5D5 08C0              <1>       or     al, al
7956 0000A5D7 75E8              <1>       jnz    short get_env_string_seq_number_repeat1
7957                            <1>
7958                            <1> get_env_string_seq_number_ok:
7959 0000A5D9 5F                <1>       pop    edi ; ***
7960 0000A5DA 89D0              <1>       mov    eax, edx ; Length of the environment string
7961                            <1>                       ; (ASCIIZ, includes ZERO tail)
7962 0000A5DC BA00300900        <1>       mov    edx, Env_Page
7963                            <1>
7964                            <1> get_env_string_stc_retn:
7965 0000A5E1 5E                <1>       pop    esi ; **
7966 0000A5E2 59                <1>       pop    ecx ; *
7967 0000A5E3 C3                <1>       retn
7968                            <1>
```

```
7969                                    <1> get_env_string_seq_number_next:
7970 0000A5E4 AC                       <1>         lodsb
7971 0000A5E5 08C0                     <1>         or      al, al
7972 0000A5E7 75FB                     <1>         jnz     short get_env_string_seq_number_next
7973                                    <1>
7974 0000A5E9 81FE00320900             <1>         cmp     esi, Env_Page + Env_Page_Size ; +512 (+4096)
7975 0000A5EF F5                       <1>         cmc
7976 0000A5F0 72EF                     <1>         jc      short get_env_string_stc_retn
7977                                    <1>
7978 0000A5F2 AC                       <1>         lodsb
7979 0000A5F3 3C01                     <1>         cmp     al, 1
7980 0000A5F5 72EA                     <1>         jb      short get_env_string_stc_retn
7981 0000A5F7 FEC1                     <1>         inc     cl
7982 0000A5F9 EBBF                     <1>         jmp     short get_env_string_seq_number_check
7983                                    <1>
7984                                    <1> get_env_string_with_word:
7985 0000A5FB 31C9                     <1>         xor     ecx, ecx
7986                                    <1>
7987                                    <1> get_env_string_calc_word_length:
7988 0000A5FD AC                       <1>         lodsb
7989 0000A5FE 3C20                     <1>         cmp     al, 20h
7990 0000A600 7211                     <1>         jb      short get_env_string_calc_word_length_ok
7991                                    <1>         ;inc    cx
7992 0000A602 FEC1                     <1>         inc     cl
7993                                    <1>
7994 0000A604 3C61                     <1>         cmp     al, 'a'
7995 0000A606 72F5                     <1>         jb      short get_env_string_calc_word_length
7996 0000A608 3C7A                     <1>         cmp     al, 'z'
7997 0000A60A 77F1                     <1>         ja      short get_env_string_calc_word_length
7998 0000A60C 24DF                     <1>         and     al, 0DFh
7999 0000A60E 8846FF                   <1>         mov     [esi-1], al
8000 0000A611 EBEA                     <1>         jmp     short get_env_string_calc_word_length
8001                                    <1>
8002                                    <1> get_env_string_calc_word_length_ok:
8003 0000A613 08C9                     <1>         or      cl, cl
8004 0000A615 7506                     <1>         jnz     short get_env_string_calc_word_length_save
8005                                    <1>
8006 0000A617 5E                       <1>         pop     esi ; **
8007                                    <1>
8008                                    <1> get_env_string_stc_retn1:
8009 0000A618 59                       <1>         pop     ecx ; *
8010                                    <1>
8011                                    <1> get_env_string_with_word_stc_retn:
8012 0000A619 31C0                     <1>         xor     eax, eax
8013 0000A61B F9                       <1>         stc
8014 0000A61C C3                       <1>         retn
8015                                    <1>
8016                                    <1> get_env_string_calc_word_length_save:
8017 0000A61D 871C24                   <1>         xchg    ebx, [esp] ; **
8018 0000A620 89DE                     <1>         mov     esi, ebx
8019                                    <1>                 ; Start of the env string (to be searched)
8020                                    <1>
8021 0000A622 57                       <1>         push    edi ; ***
8022 0000A623 89D7                     <1>         mov     edi, edx ; Env_Page
8023                                    <1>
8024                                    <1> get_env_string_compare:
8025 0000A625 57                       <1>         push    edi ; ****
8026 0000A626 51                       <1>         push    ecx ; ***** ; Variable name length
8027                                    <1>
8028                                    <1> get_env_string_compare_rep:
8029 0000A627 AC                       <1>         lodsb
8030 0000A628 AE                       <1>         scasb
8031 0000A629 7511                     <1>         jne     short get_env_string_compare_next1
8032 0000A62B E2FA                     <1>         loop    get_env_string_compare_rep
8033                                    <1>
8034 0000A62D 803F3D                   <1>         cmp     byte [edi], '='
8035 0000A630 750A                     <1>         jne     short get_env_string_compare_next1
8036                                    <1>
8037 0000A632 59                       <1>         pop     ecx ; *****
8038 0000A633 5F                       <1>         pop     edi ; ****
8039 0000A634 89FE                     <1>         mov     esi, edi
8040 0000A636 5F                       <1>         pop     edi ; ***
8041 0000A637 871C24                   <1>         xchg    ebx, [esp] ; **
8042 0000A63A EB82                     <1>         jmp     short get_env_string_move_to_buff
8043                                    <1>
8044                                    <1> get_env_string_compare_next1:
8045 0000A63C 89FE                     <1>         mov     esi, edi
8046 0000A63E 59                       <1>         pop     ecx ; *****
8047 0000A63F 5F                       <1>         pop     edi ; ****
8048                                    <1> get_env_string_compare_next2:
8049 0000A640 81FEFF310900             <1>         cmp     esi, Env_Page + Env_Page_Size - 1 ; +511 (+4095)
8050 0000A646 7310                     <1>         jnb     short get_env_string_compare_not_ok
8051 0000A648 20C0                     <1>         and     al, al
8052 0000A64A AC                       <1>         lodsb
8053 0000A64B 75F3                     <1>         jnz     short get_env_string_compare_next2
8054 0000A64D 08C0                     <1>         or      al, al
8055 0000A64F 7407                     <1>         jz      short get_env_string_compare_not_ok
8056 0000A651 4E                       <1>         dec     esi ; 12/04/2016
8057 0000A652 89F7                     <1>         mov     edi, esi
8058 0000A654 89DE                     <1>         mov     esi, ebx
8059 0000A656 EBCD                     <1>         jmp     short get_env_string_compare
8060                                    <1>
8061                                    <1> get_env_string_compare_not_ok:
8062 0000A658 5F                       <1>         pop     edi ; ***
8063 0000A659 89DE                     <1>         mov     esi, ebx
8064 0000A65B 5B                       <1>         pop     ebx ; **
8065 0000A65C EBBA                     <1>         jmp     short get_env_string_stc_retn1
8066                                    <1>
8067                                    <1> set_environment_string:
8068                                    <1>         ; 13/04/2016
8069                                    <1>         ; 12/04/2016
8070                                    <1>         ; 11/04/2016
8071                                    <1>         ; 06/04/2016
8072                                    <1>         ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
8073                                    <1>         ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
```

```
                        <1>       ; 29/08/2011
                        <1>       ; 29/08/2011
                        <1>       ; INPUT->
                        <1>       ;      ESI = ASCIIZ environment string
                        <1>       ; OUTPUT ->
                        <1>       ;      ESI is not changed
                        <1>       ;      CF = 1 -> Could not set,
                        <1>       ;           insufficient environment space
                        <1>       ;
                        <1>       ; (EAX, EDX will be changed)
                        <1>       ;
                        <1>       ;    (EAX = Start address of the env string if > 0)
                        <1>       ;    (EDX = Environment string length)
                        <1>
0000A65E 56             <1>       push   esi ; *
                        <1>
0000A65F 31C0           <1>       xor    eax, eax
                        <1>
                        <1> set_env_chk_validation1:
0000A661 FEC4           <1>       inc    ah ; variable (string) length
0000A663 AC             <1>       lodsb
0000A664 3C3D           <1>       cmp    al, '='
0000A666 7415           <1>       je     short set_env_chk_validation2
0000A668 3C20           <1>       cmp    al, 20h
0000A66A 720F           <1>       jb     short set_env_string_stc
                        <1>
                        <1>       ; 06/04/2016
0000A66C 3C61           <1>       cmp    al, 'a'
0000A66E 72F1           <1>       jb     short set_env_chk_validation1
0000A670 3C7A           <1>       cmp    al, 'z'
0000A672 77ED           <1>       ja     short set_env_chk_validation1
0000A674 2C20           <1>       sub    al, 'a'-'A'
0000A676 8846FF         <1>       mov    [esi-1], al
0000A679 EBE6           <1>       jmp    short set_env_chk_validation1
                        <1>
                        <1> set_env_string_stc:
0000A67B 5E             <1>       pop    esi ; *
                        <1>       ;stc
0000A67C C3             <1>       retn
                        <1>
                        <1> set_env_chk_validation2:
0000A67D 51             <1>       push   ecx ; **
0000A67E 53             <1>       push   ebx ; ***
0000A67F 57             <1>       push   edi ; ****
                        <1>
                        <1>       ; 12/04/2016
0000A680 8B5C240C       <1>       mov    ebx, [esp+12]
                        <1>
                        <1> set_env_chk_validation2w:
0000A684 89F7           <1>       mov    edi, esi
0000A686 4F             <1>       dec    edi
                        <1>
0000A687 807FFF20       <1>       cmp    byte [edi-1], 20h
0000A68B 771A           <1>       ja     short set_env_chk_validation2z
                        <1>
0000A68D 56             <1>       push   esi
0000A68E 89FE           <1>       mov    esi, edi
0000A690 4E             <1>       dec    esi
                        <1>
                        <1> set_env_chk_validation2x:
0000A691 4E             <1>       dec    esi
                        <1>
0000A692 39DE           <1>       cmp    esi, ebx
0000A694 7207           <1>       jb     short set_env_chk_validation2y
                        <1>
0000A696 4F             <1>       dec    edi
                        <1>
0000A697 8A06           <1>       mov    al, [esi]
0000A699 8807           <1>       mov    [edi], al
                        <1>
0000A69B EBF4           <1>       jmp    short set_env_chk_validation2x
                        <1>
                        <1> set_env_chk_validation2y:
0000A69D 5E             <1>       pop    esi
                        <1>
                        <1>       ;mov   byte [ebx], 20h
                        <1>
0000A69E 43             <1>       inc    ebx
0000A69F 895C240C       <1>       mov    [esp+12], ebx
                        <1>
0000A6A3 FECC           <1>       dec    ah ; 13/04/2016
                        <1>
0000A6A5 EBDD           <1>       jmp    short set_env_chk_validation2w
                        <1>
                        <1> set_env_chk_validation2z:
0000A6A7 BA00300900     <1>       mov    edx, Env_Page
0000A6AC 89D7           <1>       mov    edi, edx
                        <1>
                        <1> set_env_chk_validation3:
0000A6AE AC             <1>       lodsb
0000A6AF 3C20           <1>       cmp    al, 20h
0000A6B1 74FB           <1>       je     short set_env_chk_validation3
                        <1>
0000A6B3 9C             <1>       pushf
                        <1>
                        <1>       ; 12/04/2016
                        <1> set_env_chk_validation3n:
0000A6B4 3C61           <1>       cmp    al, 'a'
0000A6B6 720C           <1>       jb     short set_env_chk_validation3c
0000A6B8 3C7A           <1>       cmp    al, 'z'
0000A6BA 7705           <1>       ja     short set_env_chk_validation3x
0000A6BC 2C20           <1>       sub    al, 'a'-'A'
0000A6BE 8846FF         <1>       mov    [esi-1], al
                        <1>
                        <1> set_env_chk_validation3x:
```

```
8179 0000A6C1 AC                <1>       lodsb
8180 0000A6C2 EBF0              <1>       jmp    short set_env_chk_validation3n
8181                            <1>
8182                            <1> set_env_chk_validation3c:
8183 0000A6C4 3C20              <1>       cmp    al, 20h
8184 0000A6C6 73F9              <1>       jnb    short set_env_chk_validation3x
8185                            <1>
8186 0000A6C8 803F00            <1>       cmp    byte [edi], 0
8187 0000A6CB 7731              <1>       ja     short set_env_chk_validation4
8188                            <1>
8189 0000A6CD 9D                <1>       popf
8190 0000A6CE 7228              <1>       jb     short set_env_string_nothing
8191                            <1>
8192 0000A6D0 B900020000        <1>       mov    ecx, Env_Page_Size ; 512 (4096)
8193                            <1>
8194 0000A6D5 89DE              <1>       mov    esi, ebx ; 12/04/2016
8195                            <1>
8196                            <1> set_env_string_copy_to_envb:
8197 0000A6D7 AC                <1>       lodsb
8198 0000A6D8 3C20              <1>       cmp    al, 20h
8199 0000A6DA 720A              <1>       jb     short set_env_string_copy_to_envb_z
8200 0000A6DC AA                <1>       stosb
8201 0000A6DD E2F8              <1>       loop   set_env_string_copy_to_envb
8202                            <1>
8203                            <1>       ; 11/04/2016
8204 0000A6DF 89D7              <1>       mov    edi, edx ; Env_Page
8205 0000A6E1 B900020000        <1>       mov    ecx, Env_Page_Size
8206                            <1>
8207                            <1> set_env_string_copy_to_envb_z:
8208 0000A6E6 52                <1>       push   edx  ; Start address of the variable
8209 0000A6E7 BA00020000        <1>       mov    edx, Env_Page_Size
8210 0000A6EC 29CA              <1>       sub    edx, ecx ; variable (string) length
8211                            <1>
8212 0000A6EE 28C0              <1>       sub    al, al ; 0
8213 0000A6F0 F3AA              <1>       rep    stosb ; clear remain bytes of the env page
8214                            <1>
8215 0000A6F2 58                <1>       pop    eax  ; Start address of the variable
8216                            <1>
8217                            <1> set_env_string_allocate_envb_retn:  ; stc or clc return
8218 0000A6F3 5F                <1>       pop    edi ; ****
8219 0000A6F4 5B                <1>       pop    ebx ; ***
8220 0000A6F5 59                <1>       pop    ecx ; **
8221 0000A6F6 5E                <1>       pop    esi ; *
8222 0000A6F7 C3                <1>       retn
8223                            <1>
8224                            <1> set_env_string_nothing:
8225 0000A6F8 31C0              <1>       xor    eax, eax
8226 0000A6FA 31D2              <1>       xor    edx, edx ; 11/04/2016
8227 0000A6FC EBF5              <1>       jmp    short set_env_string_allocate_envb_retn
8228                            <1>
8229                            <1> set_env_chk_validation4:
8230                            <1>       ; 11/04/2016
8231 0000A6FE 9D                <1>       popf
8232                            <1>
8233 0000A6FF 89D6              <1>       mov    esi, edx  ; Env_Page
8234                            <1>
8235                            <1> set_env_chk_validation5:
8236 0000A701 89DF              <1>       mov    edi, ebx  ; ASCIIZ environment string address
8237 0000A703 0FB6CC            <1>       movzx  ecx, ah ; Variable (string) length (with '=')
8238                            <1>
8239                            <1> set_env_chk_validation5_loop:
8240 0000A706 AC                <1>       lodsb
8241 0000A707 AE                <1>       scasb
8242 0000A708 750A              <1>       jne    short set_env_chk_validation6
8243 0000A70A E2FA              <1>       loop   set_env_chk_validation5_loop
8244                            <1>
8245 0000A70C 3C3D              <1>       cmp    al, '='
8246 0000A70E 0F8483000000      <1>         je      set_env_change_variable
8247                            <1>
8248                            <1> set_env_chk_validation6:
8249 0000A714 08C0              <1>       or     al, al ; 0
8250 0000A716 7403              <1>       jz     short set_env_chk_validation7
8251                            <1>
8252 0000A718 AC                <1>       lodsb
8253 0000A719 EBF9              <1>       jmp    short set_env_chk_validation6
8254                            <1>
8255                            <1> set_env_chk_validation7:
8256 0000A71B 88E1              <1>       mov    cl, ah
8257 0000A71D 01F1              <1>       add    ecx, esi
8258 0000A71F 81F9FF310900      <1>       cmp    ecx, Env_Page + Env_Page_Size - 1
8259                            <1>              ; 511 (4095)
8260                            <1>              ; strlen + '=' + 0
8261 0000A725 72DA              <1>       jb     short set_env_chk_validation5
8262                            <1>
8263                            <1> set_env_chk_validation8: ; variable not found
8264 0000A727 0FB6F4            <1>       movzx  esi, ah  ; variable name length (with '=')
8265 0000A72A 01DE              <1>       add    esi, ebx ; position just after of the '='
8266                            <1>
8267                            <1> set_env_chk_validation8_loop:
8268 0000A72C AC                <1>       lodsb
8269 0000A72D 3C20              <1>       cmp    al, 20h
8270 0000A72F 74FB              <1>       je     short set_env_chk_validation8_loop
8271 0000A731 72C5              <1>       jb     short set_env_string_nothing
8272                            <1>
8273                            <1> set_env_chk_validation9:
8274 0000A733 AC                <1>       lodsb
8275 0000A734 3C20              <1>       cmp    al, 20h
8276 0000A736 73FB              <1>       jnb    short set_env_chk_validation9
8277                            <1>
8278                            <1>       ; End of ASCIIZ environment string
8279                            <1>
8280                            <1> set_env_add_variable:
8281 0000A738 29DE              <1>       sub    esi, ebx ; variable+definition length
8282                            <1>
8283 0000A73A 56                <1>       push   esi ; *****
```

```
8284                                <1>
8285 0000A73B 89D6                 <1>       mov    esi, edx ; Environment page address
8286                                <1>
8287 0000A73D B900020000           <1>       mov    ecx, Env_Page_Size ; 512 (4096)
8288                                <1>
8289                                <1> set_env_add_variable_loop:
8290 0000A742 AC                   <1>       lodsb
8291 0000A743 20C0                 <1>       and    al, al
8292 0000A745 7406                 <1>       jz     short set_env_add_variable_chk1 ; 0
8293 0000A747 E2F9                 <1>       loop   set_env_add_variable_loop
8294                                <1>
8295                                <1>       ; 11/04/2016
8296 0000A749 884EFF               <1>       mov    [esi-1], cl ; 0
8297 0000A74C 41                   <1>       inc    ecx
8298                                <1>
8299                                <1> set_env_add_variable_chk1:
8300 0000A74D 49                   <1>       dec    ecx
8301 0000A74E 7408                 <1>       jz     short set_env_add_variable_nspc
8302 0000A750 AC                   <1>       lodsb
8303 0000A751 08C0                 <1>       or     al, al
8304 0000A753 740C                 <1>       jz     short set_env_add_variable_chk2 ; 00
8305 0000A755 49                   <1>       dec    ecx
8306 0000A756 75EA                 <1>       jnz    short set_env_add_variable_loop
8307                                <1>
8308                                <1> set_env_add_variable_nspc: ; no space on environment page
8309 0000A758 58                   <1>       pop    eax ; *****
8310 0000A759 B808000000           <1>       mov    eax, 8 ; No space for new environment string
8311 0000A75E F9                   <1>       stc
8312 0000A75F EB92                 <1>         jmp     short set_env_string_allocate_envb_retn
8313                                <1>
8314                                <1> set_env_add_variable_chk2:
8315 0000A761 8B0C24               <1>       mov    ecx, [esp] ; *****
8316 0000A764 4E                   <1>       dec    esi ; beginning address of the new variable
8317 0000A765 89F0                 <1>       mov    eax, esi
8318 0000A767 01C8                 <1>       add    eax, ecx ; string length (with CR)
8319 0000A769 81C200020000         <1>       add    edx, Env_Page_Size ; 512 (4096)
8320 0000A76F 39D0                 <1>       cmp    eax, edx
8321 0000A771 77E5                 <1>       ja     short set_env_add_variable_nspc
8322 0000A773 49                   <1>       dec    ecx ; except CR at the end
8323 0000A774 89CA                 <1>       mov    edx, ecx ; 12/04/2016
8324 0000A776 89F7                 <1>       mov    edi, esi
8325 0000A778 893C24               <1>       mov    [esp], edi ; ***** ; Start address of new variable
8326 0000A77B 89DE                 <1>       mov    esi, ebx ; ASCIIZ environment string address
8327 0000A77D F3A4                 <1>       rep    movsb
8328 0000A77F 28C0                 <1>       sub    al, al
8329 0000A781 AA                   <1>       stosb
8330 0000A782 58                   <1>       pop    eax ; ***** ; Beginning address of new variable
8331 0000A783 81FF00320900         <1>         cmp     edi, Env_Page + Env_Page_Size ; 12/04/2016
8332 0000A789 0F8364FFFFFF         <1>         jnb     set_env_string_allocate_envb_retn ; OK !
8333 0000A78F 880F                 <1>       mov    [edi], cl ; 0
8334 0000A791 F8                   <1>       clc    ; 13/04/2016
8335 0000A792 E95CFFFFFF           <1>         jmp     set_env_string_allocate_envb_retn ; OK !
8336                                <1>
8337                                <1> set_env_change_variable:
8338                                <1>       ; 06/04/2016
8339                                <1>       ; esi = Variable's address in environment page (after '=')
8340                                <1>       ; edi = ASCIIZ environment string address (after '=')
8341                                <1>
8342                                <1>       ; ah = variable length from start to the '='
8343 0000A797 8825[20950100]        <1>       mov    [env_var_length], ah
8344                                <1>
8345 0000A79D 28C9                 <1>       sub    cl, cl ; ecx = 0
8346                                <1>
8347 0000A79F 57                   <1>       push   edi ; *****
8348                                <1>
8349 0000A7A0 89F7                 <1>       mov    edi, esi ; 11/04/2016
8350                                <1>
8351                                <1> set_env_change_variable_calc1:
8352 0000A7A2 AC                   <1>       lodsb
8353 0000A7A3 08C0                 <1>       or     al, al
8354 0000A7A5 7403                 <1>       jz     short set_env_change_variable_calc2
8355                                <1>
8356 0000A7A7 41                   <1>       inc    ecx ; length of environment string (after the '=')
8357                                <1>
8358 0000A7A8 EBF8                 <1>       jmp    short set_env_change_variable_calc1
8359                                <1>
8360                                <1> set_env_change_variable_calc2:
8361 0000A7AA 8B3424               <1>       mov    esi, [esp] ; ASCIIZ environment string address
8362                                <1>
8363 0000A7AD 29D2                 <1>       sub    edx, edx
8364                                <1>
8365                                <1> set_env_change_variable_calc3:
8366 0000A7AF AC                   <1>       lodsb
8367 0000A7B0 3C20                 <1>       cmp    al, 20h
8368 0000A7B2 7203                 <1>       jb     short set_env_change_variable_calc4
8369                                <1>
8370 0000A7B4 42                   <1>       inc    edx ; length of ASCIIZ string (after the '=')
8371                                <1>
8372 0000A7B5 EBF8                 <1>       jmp    short set_env_change_variable_calc3
8373                                <1>
8374                                <1> set_env_change_variable_calc4:
8375 0000A7B7 C646FF00             <1>       mov    byte [esi-1], 0  ; put ZERO instead of CR
8376                                <1>
8377 0000A7BB 5E                   <1>       pop    esi ; ***** ; ASCIIZ string address (after '=')
8378                                <1>
8379                                <1>       ; EDI = Old variable's address (after '=')
8380                                <1>
8381                                <1>       ; compare the new string with the old string
8382 0000A7BC 39CA                 <1>       cmp    edx, ecx
8383 0000A7BE 7717                 <1>       ja     short set_env_change_variable_calc5 ; longer
8384 0000A7C0 0F828F000000         <1>         jb      set_env_change_variable_calc9 ; shorter
8385                                <1>
8386                                <1>       ;same length (simple copy)
8387 0000A7C6 0FB6C4               <1>       movzx  eax, ah
8388 0000A7C9 01C2                 <1>       add    edx, eax
```

```
8389 0000A7CB F7D8              <1>       neg    eax
8390 0000A7CD 01F8              <1>       add    eax, edi
8391                            <1>       ; EAX = Start address of the variable
8392                            <1>       ; EDX = Variable length (without ZERO at the end of variable)
8393                            <1>
8394 0000A7CF F3A4              <1>       rep    movsb
8395 0000A7D1 F8                <1>       clc    ; 13/04/2016
8396 0000A7D2 E91CFFFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
8397                            <1>
8398                            <1> set_env_change_variable_calc5:
8399                            <1>       ; 11/04/2016
8400 0000A7D7 52                <1>       push   edx ; *****
8401 0000A7D8 29CA              <1>       sub    edx, ecx ; difference ; (the new string is longer)
8402 0000A7DA 89F3              <1>       mov    ebx, esi
8403 0000A7DC 89FE              <1>       mov    esi, edi
8404                            <1>
8405                            <1> set_env_change_variable_calc6:
8406 0000A7DE AC                <1>       lodsb
8407 0000A7DF 20C0              <1>       and    al, al
8408 0000A7E1 75FB              <1>       jnz    short set_env_change_variable_calc6
8409                            <1>
8410 0000A7E3 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
8411 0000A7E9 0F8369FFFFFF      <1>         jnb     set_env_add_variable_nspc
8412                            <1>
8413 0000A7EF 89F9              <1>       mov    ecx, edi  ; current (old) variable's address
8414 0000A7F1 89F7              <1>       mov    edi, esi  ; next variable's address
8415                            <1>
8416 0000A7F3 AC                <1>       lodsb
8417 0000A7F4 08C0              <1>       or     al, al
8418 0000A7F6 7416              <1>       jz     short set_env_change_variable_calc8 ; 00
8419                            <1>
8420                            <1> set_env_change_variable_calc7:
8421 0000A7F8 AC                <1>       lodsb
8422 0000A7F9 20C0              <1>       and    al, al
8423 0000A7FB 75FB              <1>       jnz    short set_env_change_variable_calc7
8424                            <1>
8425 0000A7FD 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
8426 0000A803 0F834FFFFFFF      <1>         jnb     set_env_add_variable_nspc
8427                            <1>
8428 0000A809 AC                <1>       lodsb
8429 0000A80A 08C0              <1>       or     al, al
8430 0000A80C 75EA              <1>       jnz    short set_env_change_variable_calc7
8431                            <1>
8432                            <1> set_env_change_variable_calc8:
8433 0000A80E 4E                <1>       dec    esi ; address of the second (last) 0 of the 00
8434                            <1>
8435 0000A80F 01F2              <1>       add    edx, esi ; final position of the last 0
8436                            <1>
8437 0000A811 81FA00320900      <1>       cmp    edx, Env_Page + Env_Page_Size ; 512 (4096)
8438 0000A817 0F833BFFFFFF      <1>         jnb     set_env_add_variable_nspc
8439                            <1>
8440 0000A81D 89C8              <1>       mov    eax, ecx ; old variable's address (after '=')
8441                            <1>
8442 0000A81F 89F1              <1>       mov    ecx, esi
8443 0000A821 29F9              <1>       sub    ecx, edi ; count of bytes to move forward
8444                            <1>
8445                            <1>       ; 13/04/2016
8446 0000A823 C60200            <1>       mov    byte [edx], 0
8447 0000A826 89D7              <1>       mov    edi, edx
8448 0000A828 29F2              <1>       sub    edx, esi ; difference (additional byte count)
8449 0000A82A 4F                <1>       dec    edi ; the last zero address (first byte of the 00)
8450 0000A82B 89FE              <1>       mov    esi, edi
8451 0000A82D 29D6              <1>       sub    esi, edx ; - displacement
8452                            <1>
8453 0000A82F FA                <1>       cli    ; disable interrupts
8454 0000A830 FD                <1>       std    ; backward
8455                            <1>
8456 0000A831 F3A4              <1>       rep    movsb ; move ECX bytes from DS:ESI to ES:EDI
8457                            <1>
8458 0000A833 FC                <1>       cld    ; forward (default)
8459 0000A834 FB                <1>       sti    ; enable interrupts
8460                            <1>
8461 0000A835 89C7              <1>       mov    edi, eax
8462 0000A837 59                <1>       pop    ecx ; ***** ; byte count (after '=')
8463 0000A838 89CA              <1>       mov    edx, ecx
8464 0000A83A 89DE              <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
8465 0000A83C 89FB              <1>       mov    ebx, edi
8466                            <1>
8467 0000A83E F3A4              <1>       rep    movsb
8468                            <1>
8469 0000A840 880F              <1>       mov    [edi], cl ; 0 ; end of variable
8470                            <1>
8471 0000A842 0FB605[20950100]  <1>       movzx  eax, byte [env_var_length]
8472 0000A849 01C2              <1>       add    edx, eax ; variable length (total)
8473 0000A84B F7D8              <1>       neg    eax
8474 0000A84D 01D8              <1>       add    eax, ebx ; start address of the variable
8475 0000A84F F8                <1>       clc    ; 13/04/2016
8476 0000A850 E99EFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
8477                            <1>
8478                            <1> set_env_change_variable_calc9:
8479                            <1>       ; 11/04/2016
8480 0000A855 21D2              <1>       and    edx, edx ; is empty ?
8481 0000A857 753B              <1>       jnz    short set_env_change_variable_calc15
8482                            <1>
8483 0000A859 0FB6DC            <1>       movzx  ebx, ah
8484 0000A85C F7DB              <1>       neg    ebx
8485 0000A85E 01FB              <1>       add    ebx, edi
8486                            <1>
8487                            <1>       ; EBX = Start address of the variable (in env page)
8488                            <1>       ; EDX = Variable length = 0
8489                            <1>
8490 0000A860 89FE              <1>       mov    esi, edi
8491                            <1>
8492                            <1> set_env_change_variable_calc10:
8493 0000A862 AC                <1>       lodsb
```

```
8494 0000A863 08C0              <1>       or    al, al
8495 0000A865 75FB              <1>       jnz   short set_env_change_variable_calc10
8496                            <1>
8497 0000A867 B9FF310900        <1>       mov   ecx, Env_Page + Env_Page_Size - 1
8498                            <1>
8499 0000A86C 39CE              <1>       cmp   esi, ecx ; +511 (+4095)
8500 0000A86E 7604              <1>       jna   short set_env_change_variable_calc11
8501                            <1>
8502 0000A870 89CE              <1>       mov   esi, ecx
8503 0000A872 8806              <1>       mov   [esi], al ; 0
8504                            <1>
8505                            <1> set_env_change_variable_calc11:
8506 0000A874 89DF              <1>       mov   edi, ebx ; old variable's start address
8507                            <1>
8508                            <1> set_env_change_variable_calc12:
8509 0000A876 AC                <1>       lodsb
8510 0000A877 AA                <1>       stosb
8511 0000A878 20C0              <1>       and   al, al
8512 0000A87A 75FA              <1>       jnz   short set_env_change_variable_calc12
8513 0000A87C 39CE              <1>       cmp   esi, ecx
8514 0000A87E 7706              <1>       ja    short set_env_change_variable_calc13
8515 0000A880 AC                <1>       lodsb
8516 0000A881 AA                <1>       stosb
8517 0000A882 20C0              <1>       and   al, al
8518 0000A884 75F0              <1>       jnz   short set_env_change_variable_calc12
8519                            <1>
8520                            <1> set_env_change_variable_calc13:
8521 0000A886 29F9              <1>       sub   ecx, edi
8522 0000A888 7203              <1>       jb    short set_env_change_variable_calc14
8523 0000A88A 41                <1>       inc   ecx ; 1-512 (1-4096)
8524 0000A88B F3AA              <1>       rep   stosb ; al = 0
8525                            <1>
8526                            <1> set_env_change_variable_calc14:
8527 0000A88D 29C0              <1>       sub   eax, eax ; Start address of the variable
8528                            <1>       ; EAX = 0 -> Variable is removed
8529                            <1>       ; EDX = Variable length = 0
8530                            <1>
8531 0000A88F E95FFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
8532                            <1>
8533                            <1> set_env_change_variable_calc15:
8534 0000A894 52                <1>       push  edx ; *****
8535 0000A895 F7DA              <1>       neg   edx
8536 0000A897 01CA              <1>       add   edx, ecx ; difference (the old string is longer)
8537 0000A899 89F3              <1>       mov   ebx, esi
8538 0000A89B 89FE              <1>       mov   esi, edi
8539                            <1>
8540                            <1> set_env_change_variable_calc16:
8541 0000A89D AC                <1>       lodsb
8542 0000A89E 20C0              <1>       and   al, al
8543 0000A8A0 75FB              <1>       jnz   short set_env_change_variable_calc16
8544                            <1>
8545 0000A8A2 B900320900        <1>       mov   ecx, Env_Page + Env_Page_Size
8546                            <1>
8547 0000A8A7 39CE              <1>       cmp   esi, ecx ; +512 (+4096)
8548 0000A8A9 7605              <1>       jna   short set_env_change_variable_calc17
8549                            <1>
8550 0000A8AB 89CE              <1>       mov   esi, ecx
8551 0000A8AD 8846FF            <1>       mov   [esi-1], al ; 0
8552                            <1>
8553                            <1> set_env_change_variable_calc17:
8554 0000A8B0 89F9              <1>       mov   ecx, edi  ; current (old) variable's address
8555 0000A8B2 89F7              <1>       mov   edi, esi  ; next variable's address
8556                            <1>
8557 0000A8B4 AC                <1>       lodsb
8558 0000A8B5 08C0              <1>       or    al, al
8559 0000A8B7 741D              <1>       jz    short set_env_change_variable_calc20
8560                            <1>
8561                            <1> set_env_change_variable_calc18:
8562 0000A8B9 AC                <1>       lodsb
8563 0000A8BA 20C0              <1>       and   al, al
8564 0000A8BC 75FB              <1>       jnz   short set_env_change_variable_calc18
8565                            <1>
8566 0000A8BE 81FE00320900      <1>       cmp   esi, Env_Page + Env_Page_Size
8567 0000A8C4 720B              <1>       jb    short set_env_change_variable_calc19
8568 0000A8C6 740E              <1>       je    short set_env_change_variable_calc20
8569                            <1>
8570 0000A8C8 BEFF310900        <1>       mov   esi, Env_Page + Env_Page_Size - 1
8571 0000A8CD 8806              <1>       mov   [esi], al ; 0
8572 0000A8CF EB06              <1>       jmp   short set_env_change_variable_calc21
8573                            <1>
8574                            <1> set_env_change_variable_calc19:
8575 0000A8D1 AC                <1>       lodsb
8576 0000A8D2 08C0              <1>       or    al, al
8577 0000A8D4 75E3              <1>       jnz   short set_env_change_variable_calc18
8578                            <1>
8579                            <1> set_env_change_variable_calc20:
8580 0000A8D6 4E                <1>       dec   esi ; address of the second (last) 0 of the 00
8581                            <1>
8582                            <1> set_env_change_variable_calc21:
8583                            <1>       ; edx = difference (byte count)
8584                            <1>
8585 0000A8D7 89C8              <1>       mov   eax, ecx ; old variable's address (after '=')
8586                            <1>
8587 0000A8D9 89F1              <1>       mov   ecx, esi
8588 0000A8DB 29F9              <1>       sub   ecx, edi ; count of bytes to move backward
8589                            <1>
8590 0000A8DD 89FE              <1>       mov   esi, edi ; next variable's address
8591 0000A8DF 29D7              <1>       sub   edi, edx ; (displacement)
8592                            <1>
8593 0000A8E1 F3A4              <1>       rep   movsb
8594                            <1>
8595 0000A8E3 880F              <1>       mov   [edi], cl ; 0 ; 00 ; end of environment variables
8596                            <1>
8597 0000A8E5 89C7              <1>       mov   edi, eax
8598 0000A8E7 5A                <1>       pop   edx ; ***** ; byte count (after '=')
```

```
8599 0000A8E8 89D1                <1>       mov    ecx, edx
8600 0000A8EA 89DE                <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
8601 0000A8EC 89FB                <1>       mov    ebx, edi
8602                              <1>
8603 0000A8EE F3A4                <1>       rep    movsb
8604                              <1>
8605 0000A8F0 880F                <1>       mov    [edi], cl ; 0 ; end of variable
8606                              <1>
8607 0000A8F2 0FB605[20950100]    <1>       movzx  eax, byte [env_var_length]
8608 0000A8F9 01C2                <1>       add    edx, eax ; variable length (total)
8609 0000A8FB F7D8                <1>       neg    eax
8610 0000A8FD 01D8                <1>       add    eax, ebx ; start address of the variable
8611 0000A8FF F8                  <1>       clc    ; 13/04/2016
8612 0000A900 E9EEFDFFFF          <1>         jmp     set_env_string_allocate_envb_retn ; OK !
8613                              <1>
8614                              <1> mainprog_startup_configuration:
8615                              <1>       ; 22/11/2017
8616                              <1>       ; 06/05/2016
8617                              <1>       ; 14/04/2016 (TRDOS 386 = TRDOS v2.0)
8618                              <1>       ; 17/09/2011 (TRDOS v1, MAINPROG.ASM)
8619                              <1>       ;
8620                              <1> loc_load_mainprog_cfg_file:
8621 0000A905 BE[C93F0100]        <1>       mov    esi, MainProgCfgFile
8622 0000A90A 66B80018            <1>       mov    ax, 1800h ; Except volume label and dirs
8623 0000A90E E83EEAFFFF          <1>       call   find_first_file
8624 0000A913 7256                <1>       jc     short loc_load_mainprog_cfg_exit
8625                              <1>
8626                              <1>       ;or    eax, eax
8627                              <1>       ;jz    short loc_load_mainprog_cfg_exit
8628                              <1>
8629                              <1> loc_start_mainprog_configuration:
8630                              <1>       ; ESI = FindFile_DirEntry Location
8631                              <1>       ; EAX = File Size
8632                              <1>
8633 0000A915 A3[A4890100]        <1>       mov    [MainProgCfg_FileSize], eax
8634                              <1>
8635 0000A91A 668B5614            <1>       mov    dx, [esi+DirEntry_FstClusHI]
8636 0000A91E C1E210              <1>       shl    edx, 16
8637 0000A921 668B561A            <1>       mov    dx, [esi+DirEntry_FstClusLO]
8638 0000A925 8915[D4940100]      <1>       mov    [csftdf_sf_cluster], edx
8639                              <1>
8640 0000A92B 89C1                <1>       mov    ecx, eax
8641 0000A92D 29C0                <1>       sub    eax, eax
8642                              <1>
8643                              <1>       ; TRDOS 386 (TRDOS v2.0)
8644                              <1>       ; Allocate contiguous memory block for loading the file
8645                              <1>
8646                              <1>       ; eax = 0 (Allocate memory from the beginning)
8647                              <1>       ; ecx = File (Allocation) size in bytes
8648                              <1>
8649 0000A92F E81DBAFFFF          <1>       call   allocate_memory_block
8650 0000A934 7235                <1>       jc     short loc_load_mainprog_cfg_exit
8651                              <1>
8652 0000A936 A3[CC940100]        <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
8653 0000A93B 890D[D0940100]      <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
8654                              <1>
8655 0000A941 31DB                <1>       xor    ebx, ebx
8656                              <1>       ;mov   [csftdf_sf_rbytes], ebx ; 0, reset
8657                              <1>
8658 0000A943 8A3D[B6890100]      <1>       mov    bh, [Current_Drv] ; [FindFile_Drv]
8659 0000A949 BE00010900          <1>       mov    esi, Logical_DOSDisks
8660 0000A94E 01DE                <1>       add    esi, ebx
8661                              <1>
8662 0000A950 8B1D[CC940100]      <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
8663                              <1>
8664 0000A956 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
8665 0000A95A 7710                <1>         ja   short loc_mcfg_load_fat_file
8666                              <1>
8667 0000A95C C705[DC940100]0000- <1>       mov    dword [csftdf_r_size], 65536
8667 0000A964 0100                <1>
8668 0000A966 E9A1010000          <1>         jmp     loc_mcfg_load_fs_file
8669                              <1>
8670                              <1> loc_load_mainprog_cfg_exit:
8671 0000A96B C3                  <1>       retn
8672                              <1>
8673                              <1> loc_mcfg_load_fat_file:
8674 0000A96C 0FB74611            <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
8675 0000A970 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
8676 0000A974 F7E1                <1>       mul    ecx
8677 0000A976 A3[DC940100]        <1>       mov    [csftdf_r_size], eax
8678                              <1>
8679                              <1> loc_mcfg_load_fat_file_next:
8680 0000A97B E822010000          <1>       call   mcfg_read_fat_file_sectors
8681 0000A980 0F8206010000        <1>         jc      mcfg_deallocate_mem
8682                              <1>
8683 0000A986 09D2                <1>       or     edx, edx ; edx > 0 -> EOF
8684 0000A988 74F1                <1>       jz     short loc_mcfg_load_fat_file_next
8685                              <1>
8686                              <1> loc_mcfg_load_fat_file_ok:
8687                              <1>       ; 06/05/2016
8688 0000A98A C705[70950100]-     <1>       mov    dword [mainprog_return_addr], loc_mcfg_ci_return_addr
8688 0000A990 [4DAA0000]          <1>
8689                              <1>       ;
8690 0000A994 8B35[CC940100]      <1>       mov    esi, [csftdf_sf_mem_addr]
8691 0000A99A 8935[A8890100]      <1>       mov    [MainProgCfg_LineOffset], esi
8692                              <1>
8693 0000A9A0 A1[A4890100]        <1>       mov    eax, [MainProgCfg_FileSize]
8694 0000A9A5 89C2                <1>       mov    edx, eax
8695 0000A9A7 01F2                <1>       add    edx, esi
8696                              <1>
8697                              <1> loc_mcfg_process_next_line_check:
8698 0000A9A9 89C1                <1>       mov    ecx, eax
8699                              <1>
8700 0000A9AB 803E2A              <1>       cmp    byte [esi], "*" ; Remark sign
8701 0000A9AE 7503                <1>       jne    short loc_mcfg_process_next_line
```

```
8702 0000A9B0 46                <1>       inc    esi
8703 0000A9B1 EB17              <1>       jmp    short loc_move_mainprog_cfg_nl1
8704                            <1>
8705                            <1> loc_mcfg_process_next_line:
8706 0000A9B3 83F94F            <1>       cmp    ecx, 79
8707 0000A9B6 7605              <1>       jna    short loc_start_mainprog_cfg_process
8708                            <1>
8709 0000A9B8 B94F000000        <1>       mov    ecx, 79
8710                            <1>
8711                            <1> loc_start_mainprog_cfg_process:
8712 0000A9BD BF[668A0100]      <1>       mov    edi, CommandBuffer
8713                            <1>
8714                            <1> loc_move_mainprog_cfg_line:
8715 0000A9C2 AC                <1>       lodsb
8716 0000A9C3 3C20              <1>       cmp    al, 20h
8717 0000A9C5 720C              <1>       jb     short loc_move_mainprog_cfg_nl2
8718 0000A9C7 AA                <1>       stosb
8719 0000A9C8 E2F8              <1>       loop   loc_move_mainprog_cfg_line
8720                            <1>
8721                            <1> loc_move_mainprog_cfg_nl1:
8722 0000A9CA 39D6              <1>       cmp    esi, edx ; + configuration file size
8723 0000A9CC 7312              <1>       jnb    short loc_end_of_mainprog_cfg_line
8724 0000A9CE AC                <1>       lodsb
8725 0000A9CF 3C20              <1>       cmp    al, 20h
8726 0000A9D1 73F7              <1>       jnb    short loc_move_mainprog_cfg_nl1
8727                            <1>
8728                            <1> loc_move_mainprog_cfg_nl2:
8729 0000A9D3 39D6              <1>       cmp    esi, edx
8730 0000A9D5 7309              <1>       jnb    short loc_end_of_mainprog_cfg_line
8731 0000A9D7 8A06              <1>       mov    al, [esi]
8732 0000A9D9 3C20              <1>       cmp    al, 20h
8733 0000A9DB 7703              <1>       ja     short loc_end_of_mainprog_cfg_line
8734 0000A9DD 46                <1>       inc    esi
8735 0000A9DE EBF3              <1>       jmp    short loc_move_mainprog_cfg_nl2
8736                            <1>
8737                            <1> loc_end_of_mainprog_cfg_line:
8738 0000A9E0 C60700            <1>       mov    byte [edi], 0
8739                            <1>
8740 0000A9E3 8935[A8890100]    <1>       mov    [MainProgCfg_LineOffset], esi
8741                            <1>
8742                            <1>       ; 22/11/2017
8743 0000A9E9 BE[6E8A0100]      <1>       mov    esi, CommandBuffer + 8
8744 0000A9EE 29FE              <1>       sub    esi, edi
8745 0000A9F0 7606              <1>       jna    short loc_move_mainprog_cfg_command
8746 0000A9F2 30C0              <1>       xor    al, al
8747                            <1> loc_mainprog_cfg_clear_chrs:
8748 0000A9F4 AA                <1>       stosb
8749 0000A9F5 4E                <1>       dec    esi
8750 0000A9F6 75FC              <1>       jnz    short loc_mainprog_cfg_clear_chrs
8751                            <1>
8752                            <1> loc_move_mainprog_cfg_command:
8753 0000A9F8 BE[668A0100]      <1>       mov    esi, CommandBuffer
8754 0000A9FD 89F7              <1>       mov    edi, esi
8755 0000A9FF 31DB              <1>       xor    ebx, ebx
8756                            <1>       ;xor   ecx, ecx
8757 0000AA01 30C9              <1>       xor    cl, cl
8758                            <1>
8759                            <1> loc_move_mcfg_first_cmd_char:
8760 0000AA03 8A041E            <1>       mov    al, [esi+ebx]
8761 0000AA06 FEC3              <1>       inc    bl
8762 0000AA08 3C20              <1>       cmp    al, 20h
8763 0000AA0A 7712              <1>       ja     short loc_move_mcfg_cmd_capitalizing
8764 0000AA0C 7237              <1>       jb     short loc_move_mcfg_cmd_arguments_ok
8765 0000AA0E 80FB4F            <1>       cmp    bl, 79
8766 0000AA11 72F0              <1>       jb     short loc_move_mcfg_first_cmd_char
8767 0000AA13 EB30              <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
8768                            <1>
8769                            <1> loc_move_mcfg_next_cmd_char:
8770 0000AA15 8A041E            <1>       mov    al, [esi+ebx]
8771 0000AA18 FEC3              <1>       inc    bl
8772 0000AA1A 3C20              <1>       cmp    al, 20h
8773 0000AA1C 7614              <1>       jna    short loc_move_mcfg_cmd_ok
8774                            <1>
8775                            <1> loc_move_mcfg_cmd_capitalizing:
8776 0000AA1E 3C61              <1>       cmp    al, 61h ; 'a'
8777 0000AA20 7206              <1>       jb     short loc_move_mcfg_cmd_caps_ok
8778 0000AA22 3C7A              <1>       cmp    al, 7Ah ; 'z'
8779 0000AA24 7702              <1>       ja     short loc_move_mcfg_cmd_caps_ok
8780 0000AA26 24DF              <1>       and    al, 0DFh ; sub      al, 'a'-'A'
8781                            <1>
8782                            <1> loc_move_mcfg_cmd_caps_ok:
8783 0000AA28 AA                <1>       stosb
8784 0000AA29 FEC1              <1>       inc    cl
8785 0000AA2B 80FB4F            <1>       cmp    bl, 79
8786 0000AA2E 72E5              <1>       jb     short loc_move_mcfg_next_cmd_char
8787 0000AA30 EB13              <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
8788                            <1>
8789                            <1> loc_move_mcfg_cmd_ok:
8790 0000AA32 30C0              <1>       xor    al, al ; 0
8791                            <1>
8792                            <1> loc_move_mcfg_cmd_arguments:
8793 0000AA34 8807              <1>       mov    [edi], al
8794 0000AA36 47                <1>       inc    edi
8795 0000AA37 80FB4F            <1>       cmp    bl, 79
8796 0000AA3A 7309              <1>       jnb    short loc_move_mcfg_cmd_arguments_ok
8797 0000AA3C 8A041E            <1>       mov    al, [esi+ebx]
8798 0000AA3F FEC3              <1>       inc    bl
8799 0000AA41 3C20              <1>       cmp    al, 20h
8800 0000AA43 73EF              <1>       jnb    short loc_move_mcfg_cmd_arguments
8801                            <1>
8802                            <1> loc_move_mcfg_cmd_arguments_ok:
8803 0000AA45 C60700            <1>       mov    byte [edi], 0
8804                            <1>
8805                            <1> loc_mcfg_process_cmd_interpreter:
8806 0000AA48 E825E0FFFF        <1>       call    command_interpreter
```

```
8807                                    <1>
8808                                    <1> loc_mcfg_ci_return_addr:
8809 0000AA4D A1[A4890100]              <1>         mov     eax, [MainProgCfg_FileSize]
8810 0000AA52 89C2                      <1>         mov     edx, eax
8811 0000AA54 8B35[A8890100]            <1>         mov     esi, [MainProgCfg_LineOffset]
8812 0000AA5A 01F2                      <1>         add     edx, esi
8813 0000AA5C 0305[CC940100]            <1>         add     eax, [csftdf_sf_mem_addr]
8814 0000AA62 29F0                      <1>         sub     eax, esi
8815 0000AA64 0F873FFFFFFF              <1>           ja      loc_mcfg_process_next_line_check
8816                                    <1>
8817 0000AA6A E81D000000                <1>         call    mcfg_deallocate_mem
8818                                    <1>
8819 0000AA6F B94F000000                <1>         mov     ecx, 79 ; 80 ?
8820 0000AA74 BF[668A0100]              <1>         mov     edi, CommandBuffer
8821 0000AA79 30C0                      <1>         xor     al, al
8822 0000AA7B F3AA                      <1>         rep     stosb
8823                                    <1>
8824                                    <1>         ; 06/05/2016
8825 0000AA7D BE[1F4C0100]              <1>         mov     esi, nextline
8826 0000AA82 E89EC9FFFF                <1>         call    print_msg
8827 0000AA87 E963D6FFFF                <1>         jmp     dos_prompt
8828                                    <1>
8829                                    <1> mcfg_deallocate_mem:
8830 0000AA8C A1[CC940100]              <1>         mov     eax, [csftdf_sf_mem_addr] ; start address
8831 0000AA91 8B0D[D0940100]            <1>         mov     ecx, [csftdf_sf_mem_bsize] ; block size
8832                                    <1>         ;call   deallocate_memory_block
8833                                    <1>         ;retn
8834 0000AA97 E9C2BAFFFF                <1>         jmp     deallocate_memory_block
8835                                    <1>
8836                                    <1> mcfg_read_file_sectors:
8837                                    <1>         ; 14/04/2016
8838 0000AA9C 807E0300                  <1>         cmp     byte [esi+LD_FATType], 0
8839 0000AAA0 7669                      <1>           jna   short mcfg_read_fs_file_sectors
8840                                    <1>
8841                                    <1> mcfg_read_fat_file_sectors:
8842                                    <1>         ; return:
8843                                    <1>         ;   CF = 0 & EDX > 0 -> END OF FILE
8844                                    <1>         ;   CF = 0 & EDX = 0 -> not EOF
8845                                    <1>         ;   CF = 1 -> read error (error code in AL)
8846                                    <1>
8847                                    <1> mcfg_read_fat_file_secs_0:
8848 0000AAA2 8B15[A4890100]            <1>         mov     edx, [MainProgCfg_FileSize]
8849 0000AAA8 2B15[E4940100]            <1>         sub     edx, [csftdf_sf_rbytes]
8850 0000AAAE 3B15[DC940100]            <1>         cmp     edx, [csftdf_r_size]
8851 0000AAB4 7306                      <1>         jnb     short mcfg_read_fat_file_secs_1
8852 0000AAB6 8915[DC940100]            <1>         mov     [csftdf_r_size], edx
8853                                    <1>
8854                                    <1> mcfg_read_fat_file_secs_1:
8855 0000AABC A1[DC940100]              <1>         mov     eax, [csftdf_r_size]
8856 0000AAC1 29D2                      <1>         sub     edx, edx
8857 0000AAC3 0FB74E11                  <1>         movzx   ecx, word [esi+LD_BPB+BytesPerSec]
8858 0000AAC7 01C8                      <1>         add     eax, ecx
8859 0000AAC9 48                        <1>         dec     eax
8860 0000AACA F7F1                      <1>         div     ecx
8861 0000AACC 89C1                      <1>         mov     ecx, eax ; sector count
8862 0000AACE A1[D4940100]              <1>         mov     eax, [csftdf_sf_cluster]
8863                                    <1>
8864                                    <1>         ; EBX = memory block address (current)
8865                                    <1>
8866 0000AAD3 E88C230000                <1>         call    read_fat_file_sectors
8867 0000AAD8 7230                      <1>         jc      short mcfg_read_fat_file_secs_3
8868                                    <1>
8869                                    <1>         ; EBX = next memory address
8870                                    <1>
8871 0000AADA A1[E4940100]              <1>         mov     eax, [csftdf_sf_rbytes]
8872 0000AADF 0305[DC940100]            <1>         add     eax, [csftdf_r_size]
8873 0000AAE5 8B15[A4890100]            <1>         mov     edx, [MainProgCfg_FileSize]
8874 0000AAEB 39D0                      <1>         cmp     eax, edx
8875 0000AAED 731B                      <1>         jnb     short mcfg_read_fat_file_secs_3 ; edx > 0
8876 0000AAEF A3[E4940100]              <1>         mov     [csftdf_sf_rbytes], eax
8877                                    <1>
8878 0000AAF4 53                        <1>         push    ebx ; *
8879                                    <1>         ; get next cluster (csftdf_r_size! bytes)
8880 0000AAF5 A1[D4940100]              <1>         mov     eax, [csftdf_sf_cluster]
8881 0000AAFA E837210000                <1>         call    get_next_cluster
8882 0000AAFF 5B                        <1>         pop     ebx ; *
8883 0000AB00 7301                      <1>         jnc     short mcfg_read_fat_file_secs_2
8884                                    <1>
8885                                    <1>         ;mov    eax, 17; Read error !
8886 0000AB02 C3                        <1>         retn
8887                                    <1>
8888                                    <1> mcfg_read_fat_file_secs_2:
8889 0000AB03 29D2                      <1>         sub     edx, edx ; 0
8890 0000AB05 A3[D4940100]              <1>         mov     [csftdf_sf_cluster], eax ; next cluster
8891                                    <1>
8892                                    <1> mcfg_read_fat_file_secs_3:
8893 0000AB0A C3                        <1>         retn
8894                                    <1>
8895                                    <1> mcfg_read_fs_file_sectors:
8896 0000AB0B C3                        <1>         retn
8897                                    <1>
8898                                    <1> loc_mcfg_load_fs_file:
8899 0000AB0C C3                        <1>         retn
8900                                    <1>
8901                                    <1> load_and_execute_file:
8902                                    <1>         ; 04/01/2017
8903                                    <1>         ; 06/05/2016, 07/05/2016, 11/05/2016
8904                                    <1>         ; 23/04/2016, 24/04/2016
8905                                    <1>         ; 22/04/2016 (TRDOS 386 = TRDOS v2.0)
8906                                    <1>         ; 05/11/2011
8907                                    <1>         ; (TRDOS v1, CMDINTR.ASM, 'cmp_cmd_run', 'cmp_cmd_external')
8908                                    <1>         ; ('loc_run_check_filename')
8909                                    <1>         ; 29/08/2011
8910                                    <1>         ; 10/09/2011
8911                                    <1>         ; INPUT->
```

```
8912                                    <1>       ;      ESI = Path Name address (CommandBuffer address)
8913                                    <1>       ; OUTPUT ->
8914                                    <1>       ;      none (error message will be shown if an error will occur)
8915                                    <1>       ;
8916                                    <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI, EBP will be changed)
8917                                    <1>       ;
8918                                    <1> loc_run_check_filename:
8919 0000AB0D 803E20                   <1>       cmp    byte [esi], 20h
8920 0000AB10 0F822BE3FFFF             <1>       jb     loc_cmd_failed
8921 0000AB16 7703                     <1>       ja     short loc_run_check_filename_ok
8922 0000AB18 46                       <1>       inc    esi
8923 0000AB19 EBF2                     <1>       jmp    short loc_run_check_filename
8924                                    <1>
8925                                    <1> loc_run_check_filename_ok:
8926 0000AB1B C605[178A0100]00         <1>       mov    byte [CmdArgStart], 0 ; reset
8927 0000AB22 56                       <1>       push   esi ; *
8928                                    <1> loc_run_get_first_arg_pos:
8929 0000AB23 46                       <1>       inc    esi
8930 0000AB24 8A06                     <1>       mov    al, [esi]
8931 0000AB26 3C20                     <1>       cmp    al, 20h
8932 0000AB28 77F9                     <1>       ja     short loc_run_get_first_arg_pos
8933 0000AB2A C60600                   <1>       mov    byte [esi], 0
8934                                    <1> loc_run_get_external_arg_pos:
8935                                    <1>       ; 11/05/2016
8936 0000AB2D 46                       <1>       inc    esi
8937 0000AB2E 8A06                     <1>       mov    al, [esi]
8938 0000AB30 3C20                     <1>       cmp    al, 20h
8939 0000AB32 760C                     <1>       jna    short loc_run_parse_path_name
8940 0000AB34 89F0                     <1>       mov    eax, esi
8941 0000AB36 2D[668A0100]             <1>       sub    eax, CommandBuffer
8942 0000AB3B A2[178A0100]             <1>       mov    byte [CmdArgStart], al
8943                                    <1> loc_run_parse_path_name:
8944 0000AB40 5E                       <1>       pop    esi ; *
8945 0000AB41 BF[56920100]             <1>       mov    edi, FindFile_Drv
8946 0000AB46 E8D7090000               <1>       call   parse_path_name
8947 0000AB4B 0F82F0E2FFFF             <1>       jc     loc_cmd_failed
8948                                    <1>
8949                                    <1> loc_run_check_filename_exists:
8950 0000AB51 BE[98920100]             <1>       mov    esi, FindFile_Name
8951 0000AB56 803E20                   <1>       cmp    byte [esi], 20h
8952 0000AB59 0F86E2E2FFFF             <1>       jna    loc_cmd_failed
8953                                    <1>
8954                                    <1> loc_run_check_exe_filename_ext:
8955 0000AB5F E890020000               <1>       call   check_prg_filename_ext
8956 0000AB64 0F82D7E2FFFF             <1>       jc     loc_cmd_failed
8957                                    <1>
8958                                    <1> loc_run_check_exe_filename_ext_ok:
8959 0000AB6A 66A3[6E950100]           <1>       mov    word [EXE_ID], ax
8960                                    <1>
8961                                    <1> loc_run_drv:
8962 0000AB70 C605[6D950100]00         <1>       mov    byte [Run_Manual_Path], 0
8963 0000AB77 A1[B0890100]             <1>       mov    eax, [Current_Dir_FCluster]
8964 0000AB7C A3[68950100]             <1>         mov     [Run_CDirFC], eax
8965                                    <1>       ;
8966 0000AB81 8A35[B6890100]           <1>       mov    dh, [Current_Drv]
8967 0000AB87 8835[12910100]           <1>       mov    [RUN_CDRV], dh
8968                                    <1>
8969 0000AB8D 8A15[56920100]           <1>       mov    dl, [FindFile_Drv]
8970 0000AB93 38F2                     <1>       cmp    dl, dh
8971 0000AB95 7412                     <1>       je     short loc_run_change_directory
8972                                    <1>
8973 0000AB97 8005[6D950100]02         <1>       add    byte [Run_Manual_Path], 2
8974                                    <1>
8975 0000AB9E E80BD4FFFF               <1>       call   change_current_drive
8976 0000ABA3 0F82C3E2FFFF             <1>       jc     loc_run_cmd_failed
8977                                    <1>
8978                                    <1> loc_run_change_directory:
8979 0000ABA9 803D[57920100]20         <1>       cmp    byte [FindFile_Directory], 20h
8980 0000ABB0 7623                     <1>       jna    short loc_run_find_executable_file
8981                                    <1>
8982 0000ABB2 FE05[6D950100]           <1>       inc    byte [Run_Manual_Path]
8983                                    <1>
8984 0000ABB8 FE05[833F0100]           <1>       inc    byte [Restore_CDIR]
8985                                    <1>
8986 0000ABBE BE[57920100]             <1>       mov    esi, FindFile_Directory
8987 0000ABC3 30E4                     <1>       xor    ah, ah ; CD_COMMAND sign -> 0
8988 0000ABC5 E842030000               <1>       call   change_current_directory
8989 0000ABCA 0F829CE2FFFF             <1>       jc     loc_run_cmd_failed
8990                                    <1>
8991                                    <1> loc_run_change_prompt_dir_string:
8992 0000ABD0 E857020000               <1>       call   change_prompt_dir_string
8993                                    <1>
8994                                    <1> loc_run_find_executable_file:
8995 0000ABD5 66C705[6C950100]00-       <1>       mov    word [Run_Auto_Path], 0
8995 0000ABDD 00                       <1>
8996                                    <1>
8997                                    <1> loc_run_find_executable_file_next:
8998 0000ABDE BE[98920100]             <1>       mov    esi, FindFile_Name
8999                                    <1> loc_run_find_program_file_next:
9000 0000ABE3 66B80018                 <1>       mov    ax, 1800h ; Except volume label and dirs
9001 0000ABE7 E865E7FFFF               <1>       call   find_first_file
9002                                    <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
9003                                    <1>       ; EDI = Directory Buffer Directory Entry Location
9004                                    <1>       ; EAX = File size
9005 0000ABEC 0F835C010000             <1>       jnc    loc_load_and_run_file
9006                                    <1>
9007 0000ABF2 3C02                     <1>       cmp    al, 2 ; file not found
9008 0000ABF4 0F8572E2FFFF             <1>       jne    loc_run_cmd_failed
9009                                    <1>
9010 0000ABFA 66A1[6E950100]           <1>       mov    ax, word [EXE_ID]
9011 0000AC00 80FC2E                   <1>       cmp    ah, '.' ; File name has extension sign
9012 0000AC03 7424                     <1>       je     short loc_run_check_auto_path
9013                                    <1>
9014 0000AC05 08C0                     <1>       or     al, al
9015 0000AC07 7520                     <1>       jnz    short loc_run_check_auto_path
```

```
9016                                    <1>
9017 0000AC09 80FC08                    <1>       cmp    ah, 8 ; count of file name chars
9018 0000AC0C 771B                      <1>       ja     short loc_run_check_auto_path
9019                                    <1>
9020                                    <1> loc_run_change_file_ext_to_prg:
9021 0000AC0E 0FB6DC                    <1>       movzx  ebx, ah ; count of file name chars
9022 0000AC11 BE[98920100]              <1>       mov    esi, FindFile_Name
9023 0000AC16 01F3                      <1>       add    ebx, esi
9024                                    <1>       ; 07/05/2016
9025 0000AC18 C7032E505247              <1>       mov    dword [ebx],  '.PRG'
9026 0000AC1E 66C705[6E950100]50-       <1>       mov    word [EXE_ID], 'P.'
9026 0000AC26 2E                        <1>
9027 0000AC27 EBBA                      <1>       jmp    short loc_run_find_program_file_next
9028                                    <1>
9029                                    <1> loc_run_check_auto_path:
9030                                    <1>       ; NOTE: /// 07/05/2016 ///
9031                                    <1>       ; If the path is given, value of byte [Run_Manual_Path]
9032                                    <1>       ; will not be ZERO. If so, file searching by using
9033                                    <1>       ; Automatic Path (via 'PATH' environment variable)
9034                                    <1>       ; will not be applicable, because the program file
9035                                    <1>       ; is already/absolutely not found.
9036                                    <1>
9037 0000AC29 A0[6D950100]              <1>       mov    al, [Run_Manual_Path]
9038 0000AC2E 08C0                      <1>       or     al, al
9039 0000AC30 0F850BE2FFFF              <1>       jnz    loc_cmd_failed
9040                                    <1>
9041                                    <1> loc_run_check_auto_path_again:
9042 0000AC36 66833D[6C950100]FF        <1>       cmp    word [Run_Auto_Path], 0FFFFh
9043                                    <1>              ; 0FFFFh = Not a valid run path (in ENV block)
9044 0000AC3E 0F83FDE1FFFF              <1>       jnb    loc_cmd_failed
9045                                    <1>       ; xor  al, al
9046 0000AC44 BE[4F400100]              <1>       mov    esi, Cmd_Path ; 'PATH'
9047 0000AC49 BF[B68A0100]              <1>       mov    edi, TextBuffer
9048 0000AC4E E848F9FFFF                <1>       call   get_environment_string
9049 0000AC53 730E                      <1>       jnc    short loc_run_chk_filename_ext_again
9050 0000AC55 66C705[6C950100]FF-       <1>       mov    word [Run_Auto_Path], 0FFFFh ; invalid
9050 0000AC5D FF                        <1>
9051 0000AC5E E9DEE1FFFF                <1>       jmp    loc_cmd_failed
9052                                    <1>
9053                                    <1> loc_run_chk_filename_ext_again:
9054 0000AC63 89C1                      <1>       mov    ecx, eax ; string length (with zero tail)
9055 0000AC65 49                        <1>       dec    ecx ; without zero tail
9056 0000AC66 66A1[6E950100]            <1>       mov    ax, [EXE_ID]
9057 0000AC6C 80FC2E                    <1>       cmp    ah, '.'
9058 0000AC6F 740E                      <1>       je     short loc_run_chk_auto_path_pos
9059                                    <1>
9060                                    <1> loc_run_change_file_ext_to_noext_again:
9061 0000AC71 0FB6DC                    <1>       movzx  ebx, ah
9062 0000AC74 BE[98920100]              <1>       mov    esi, FindFile_Name
9063 0000AC79 01F3                      <1>       add    ebx, esi
9064 0000AC7B 29C0                      <1>       sub    eax, eax
9065 0000AC7D 8903                      <1>       mov    [ebx], eax ; 0 ; erase extension (.PRG)
9066                                    <1>
9067                                    <1> loc_run_chk_auto_path_pos:
9068                                    <1>       ;movzx eax,  word [Run_Auto_Path]
9069 0000AC7F 66A1[6C950100]            <1>       mov    ax, [Run_Auto_Path]
9070 0000AC85 39C8                      <1>       cmp    eax, ecx ; ecx = string length (except zero tail)
9071 0000AC87 0F83B4E1FFFF              <1>       jnb    loc_cmd_failed
9072                                    <1>       ;or    eax, eax
9073 0000AC8D 6609C0                    <1>       or     ax, ax
9074 0000AC90 7502                      <1>       jnz    short loc_run_auto_path_pos_move
9075 0000AC92 B005                      <1>       mov    al, 5
9076                                    <1>
9077                                    <1> loc_run_auto_path_pos_move:
9078 0000AC94 89FE                      <1>       mov    esi, edi ; offset TextBuffer
9079 0000AC96 01C6                      <1>       add    esi, eax
9080                                    <1>
9081                                    <1> loc_run_auto_path_pos_space_loop:
9082 0000AC98 AC                        <1>       lodsb
9083 0000AC99 3C20                      <1>       cmp    al, 20h
9084 0000AC9B 74FB                      <1>       je     short loc_run_auto_path_pos_space_loop
9085 0000AC9D 0F829EE1FFFF              <1>       jb     loc_cmd_failed
9086 0000ACA3 AA                        <1>       stosb
9087                                    <1> loc_run_auto_path_pos_move_next:
9088 0000ACA4 AC                        <1>       lodsb
9089 0000ACA5 3C3B                      <1>       cmp    al, ';'
9090 0000ACA7 7414                      <1>       je     short loc_run_auto_path_pos_move_last_byte
9091 0000ACA9 3C20                      <1>       cmp    al, 20h
9092 0000ACAB 74F7                      <1>       je     short loc_run_auto_path_pos_move_next
9093 0000ACAD 7203                      <1>       jb     short loc_byte_ptr_end_of_path
9094 0000ACAF AA                        <1>       stosb
9095 0000ACB0 EBF2                      <1>       jmp    short loc_run_auto_path_pos_move_next
9096                                    <1>
9097                                    <1> loc_byte_ptr_end_of_path:
9098 0000ACB2 66C705[6C950100]FF-       <1>       mov    word [Run_Auto_Path], 0FFFFh ; end of path
9098 0000ACBA FF                        <1>
9099 0000ACBB EB0D                      <1>       jmp    short loc_run_auto_path_move_ok
9100                                    <1>
9101                                    <1> loc_run_auto_path_pos_move_last_byte:
9102 0000ACBD 89F0                      <1>       mov    eax, esi
9103 0000ACBF 2D[B68A0100]              <1>       sub    eax, TextBuffer
9104 0000ACC4 66A3[6C950100]            <1>       mov    [Run_Auto_Path], ax ; next path position
9105                                    <1>
9106                                    <1> loc_run_auto_path_move_ok:
9107 0000ACCA 4F                        <1>       dec    edi
9108 0000ACCB B02F                      <1>       mov    al, '/'
9109 0000ACCD 3807                      <1>       cmp    [edi], al
9110 0000ACCF 7403                      <1>       je     short loc_run_auto_path_move_file_name
9111 0000ACD1 47                        <1>       inc    edi
9112 0000ACD2 8807                      <1>       mov    [edi], al
9113                                    <1>
9114                                    <1> loc_run_auto_path_move_file_name:
9115 0000ACD4 47                        <1>       inc    edi
9116 0000ACD5 BE[98920100]              <1>       mov    esi, FindFile_Name
9117                                    <1>
```

```
                            <1> loc_run_auto_path_move_fn_loop:
0000ACDA AC                 <1>     lodsb
0000ACDB AA                 <1>     stosb
0000ACDC 08C0               <1>     or      al, al
0000ACDE 75FA               <1>     jnz     short loc_run_auto_path_move_fn_loop
                            <1>
0000ACE0 BE[B68A0100]       <1>     mov     esi, TextBuffer
0000ACE5 BF[56920100]       <1>     mov     edi, FindFile_Drv
0000ACEA E833080000         <1>     call    parse_path_name
0000ACEF 0F824CE1FFFF       <1>     jc      loc_cmd_failed
                            <1>
0000ACF5 8A35[B6890100]     <1>     mov     dh, [Current_Drv]
0000ACFB 8A15[56920100]     <1>     mov     dl, [FindFile_Drv]
0000AD01 38F2               <1>     cmp     dl, dh
0000AD03 740B               <1>     je      short loc_run_change_directory_again
                            <1>
0000AD05 E8A4D2FFFF         <1>     call    change_current_drive
0000AD0A 0F825CE1FFFF       <1>     jc      loc_run_cmd_failed
                            <1>
                            <1> loc_run_change_directory_again:
0000AD10 803D[57920100]20   <1>     cmp     byte [FindFile_Directory], 20h
0000AD17 761D               <1>     jna     short loc_load_executable_cdir_chk_again
                            <1>
0000AD19 FE05[833F0100]     <1>     inc     byte [Restore_CDIR]
0000AD1F BE[57920100]       <1>     mov     esi, FindFile_Directory
0000AD24 30E4               <1>     xor     ah, ah ; CD_COMMAND sign -> 0
0000AD26 E8E1010000         <1>     call    change_current_directory
0000AD2B 0F823BE1FFFF       <1>     jc      loc_run_cmd_failed
                            <1>
                            <1> loc_run_chg_prompt_dir_str_again:
0000AD31 E8F6000000         <1>     call    change_prompt_dir_string
                            <1>
                            <1> loc_load_executable_cdir_chk_again:
0000AD36 A1[B0890100]       <1>     mov     eax, [Current_Dir_FCluster]
0000AD3B 3B05[68950100]     <1>     cmp     eax, [Run_CDirFC]
0000AD41 0F8597FEFFFF       <1>     jne     loc_run_find_executable_file_next
0000AD47 30C0               <1>     xor     al, al ; 0
0000AD49 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
0000AD4E BE[98920100]       <1>     mov     esi, FindFile_Name
0000AD53 BF[B68A0100]       <1>     mov     edi, TextBuffer
                            <1>
                            <1>     ; 24/04/2016
0000AD58 31D2               <1>     xor     edx, edx
0000AD5A 668915[4A040300]   <1>     mov     word [argc], dx ; 0
0000AD61 8915[8C030300]     <1>     mov     dword [u.nread], edx ; 0
                            <1>
                            <1> loc_load_and_run_file_1:
0000AD67 AC                 <1>     lodsb
0000AD68 AA                 <1>     stosb
0000AD69 FF05[8C030300]     <1>     inc     dword [u.nread]
0000AD6F 20C0               <1>     and     al, al
0000AD71 75F4               <1>     jnz     short loc_load_and_run_file_1
                            <1>
0000AD73 A0[178A0100]       <1>     mov     al, [CmdArgStart]
0000AD78 20C0               <1>     and     al, al
0000AD7A 7445               <1>     jz      short loc_load_and_run_file_7
                            <1>
0000AD7C 0FB6F0             <1>     movzx   esi, al ; 11/05/2016
0000AD7F B950000000         <1>     mov     ecx, 80
0000AD84 29F1               <1>     sub     ecx, esi
0000AD86 81C6[668A0100]     <1>     add     esi, CommandBuffer
                            <1>
0000AD8C 66FF05[4A040300]   <1>     inc     word [argc] ; 11/05/2016
                            <1>
                            <1> loc_load_and_run_file_2:
0000AD93 AC                 <1>     lodsb
0000AD94 3C20               <1>     cmp     al, 20h
0000AD96 7717               <1>     ja      short loc_load_and_run_file_5
0000AD98 721E               <1>     jb      short loc_load_and_run_file_6
                            <1>
                            <1> loc_load_and_run_file_3:
0000AD9A 803E20             <1>     cmp     byte [esi], 20h
0000AD9D 7707               <1>     ja      short loc_load_and_run_file_4
0000AD9F 7217               <1>     jb      short loc_load_and_run_file_6
0000ADA1 46                 <1>     inc     esi
0000ADA2 E2F6               <1>     loop    loc_load_and_run_file_3
0000ADA4 EB12               <1>     jmp     short loc_load_and_run_file_6
                            <1>
                            <1> loc_load_and_run_file_4:
0000ADA6 28C0               <1>     sub     al, al ; 0
0000ADA8 66FF05[4A040300]   <1>     inc     word [argc]
                            <1> loc_load_and_run_file_5:
0000ADAF AA                 <1>     stosb
0000ADB0 FF05[8C030300]     <1>     inc     dword [u.nread]
0000ADB6 E2DB               <1>     loop    loc_load_and_run_file_2
                            <1>
                            <1> loc_load_and_run_file_6:
0000ADB8 30C0               <1>     xor     al, al ; 0
0000ADBA AA                 <1>     stosb
0000ADBB FF05[8C030300]     <1>     inc     dword [u.nread]
                            <1> loc_load_and_run_file_7:
0000ADC1 8807               <1>     mov     [edi], al ; 0
0000ADC3 66FF05[4A040300]   <1>     inc     word [argc] ; 24/04/2016
0000ADCA FF05[8C030300]     <1>     inc     dword [u.nread] ; 24/04/2016
0000ADD0 BE[B68A0100]       <1>     mov     esi, TextBuffer
0000ADD5 8B15[C4920100]     <1>     mov     edx, [FindFile_DirEntry+DirEntry_FileSize]
0000ADDB 66A1[BC920100]     <1>     mov     ax, [FindFile_DirEntry+DirEntry_FstClusHI]
0000ADE1 C1E010             <1>     shl     eax, 16 ; 13/11/2017
0000ADE4 66A1[C2920100]     <1>     mov     ax, [FindFile_DirEntry+DirEntry_FstClusLO]
                            <1>     ; EAX = First Cluster number
```

```
                            <1>       ; EDX = File Size
                            <1>       ; ESI = Argument list address
                            <1>       ; [argc] = argument count
                            <1>       ; [u.nread] = argument list length
0000ADEA E85A640000         <1>       call   load_and_run_file ; trdosk6.s
                            <1>         ;jc  loc_run_cmd_failed ; 04/01/2017
                            <1> loc_load_and_run_file_8: ; 06/05/2016
0000ADEF E98BE9FFFF         <1>       jmp    loc_file_rw_restore_retn
                            <1>
                            <1> check_prg_filename_ext:
                            <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 10/09/2011
                            <1>       ; (TRDOS v1, CMDINTR.ASM, 'proc_check_exe_filename_ext')
                            <1>       ; 14/11/2009
                            <1>       ; INPUT ->
                            <1>       ;      ESI = Dot File Name
                            <1>       ; OUTPUT ->
                            <1>       ;     cf = 0 -> EXE_ID in AL
                            <1>       ;      ESI = Last char + 1 position
                            <1>       ;     cf = 1 -> Invalid executable file name
                            <1>       ;      or no file name extension if AH<=8
                            <1>       ;      AL = Last file name char
                            <1>       ;     cf = 0 -> AL='P' (PRG), AL=0 (no extension)
                            <1>       ;
                            <1>       ; (Modified registers: EAX, ESI)
                            <1>
0000ADF4 30E4               <1>       xor    ah, ah
                            <1> loc_run_check_filename_ext:
0000ADF6 AC                 <1>       lodsb
0000ADF7 3C21               <1>       cmp    al, 21h
0000ADF9 7229               <1>       jb     short loc_check_exe_fn_retn
0000ADFB FEC4               <1>       inc    ah
0000ADFD 3C2E               <1>       cmp    al, '.'
0000ADFF 75F5               <1>       jne    short loc_run_check_filename_ext
                            <1>
                            <1> loc_run_check_filename_ext_dot:
0000AE01 80FC02             <1>       cmp    ah, 2 ; .??? is not valid
0000AE04 88C4               <1>       mov    ah, al ; '.'
0000AE06 7219               <1>       jb     short loc_check_prg_fn_retn
                            <1>
                            <1> loc_run_check_filename_ext_dot_ok:
0000AE08 AC                 <1>       lodsb
0000AE09 24DF               <1>       and    al, 0DFh
                            <1>
                            <1> loc_run_check_filename_ext_prg:
0000AE0B 3C50               <1>       cmp    al, 'P'
0000AE0D 7212               <1>       jb     short loc_check_prg_fn_retn
0000AE0F 7711               <1>       ja     short loc_check_prg_fn_stc
0000AE11 AC                 <1>       lodsb
0000AE12 24DF               <1>       and    al, 0DFh
0000AE14 3C52               <1>       cmp    al, 'R'
0000AE16 750A               <1>       jne    short loc_check_prg_fn_stc
0000AE18 AC                 <1>       lodsb
0000AE19 24DF               <1>       and    al, 0DFh
0000AE1B 3C47               <1>       cmp    al, 'G'
0000AE1D 7503               <1>       jne    short loc_check_prg_fn_stc
                            <1>
0000AE1F B050               <1>       mov    al, 'P'
                            <1> loc_check_prg_fn_retn:
0000AE21 C3                 <1>       retn
                            <1>
                            <1> loc_check_prg_fn_stc:
0000AE22 F9                 <1>       stc
0000AE23 C3                 <1>       retn
                            <1>
                            <1> loc_check_exe_fn_retn:
0000AE24 28C0               <1>       sub    al, al ; 0
0000AE26 C3                 <1>       retn
                            <1>
                            <1> find_and_list_files:
0000AE27 C3                 <1>       retn
                            <1> set_exec_arguments:
0000AE28 C3                 <1>       retn
                            <1> delete_fs_directory:
0000AE29 31C0               <1>       xor eax, eax
0000AE2B C3                 <1>       retn
                                %include 'trdosk4.s' ; 24/01/2016
                            <1> ; ****************************************************************************
                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - Directory Functions : trdosk4.s
                            <1> ; ----------------------------------------------------------------------------
                            <1> ; Last Update: 02/03/2021
                            <1> ; ----------------------------------------------------------------------------
                            <1> ; Beginning: 24/01/2016
                            <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> ; DIR.ASM (09/10/2011)
                            <1> ; ****************************************************************************
                            <1>
                            <1> ; DIR.ASM  [ TRDOS KERNEL - COMMAND EXECUTER SECTION - DIRECTORY FUNCTIONS ]
                            <1> ; (c) 2004-2010  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/10/2011
                            <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
                            <1>
                            <1> change_prompt_dir_string:
                            <1>       ; 05/10/2016
                            <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 27/03/2011
                            <1>       ; 09/10/2009
                            <1>       ; INPUT/OUTPUT => none
                            <1>       ; this procedure changes current directory string/text
                            <1>       ; 2005
                            <1>
0000AE2C BE[13910100]       <1>       mov    esi, PATH_Array
                            <1> change_prompt_dir_str: ; 05/10/2016 (call from 'set_working_path')
```

```
3116 0000AE31 BF[BA890100]        <1>       mov    edi, Current_Directory
3117 0000AE36 8A25[B4890100]      <1>       mov    ah, [Current_Dir_Level]
3118 0000AE3C E807000000          <1>       call   set_current_directory_string
3119 0000AE41 880D[158A0100]      <1>       mov    [Current_Dir_StrLen], cl
3120                              <1>
3121 0000AE47 C3                  <1>       retn
3122                              <1>
3123                              <1> set_current_directory_string:
3124                              <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
3125                              <1>       ; 27/03/2011
3126                              <1>       ; 09/10/2009
3127                              <1>       ; INPUT:
3128                              <1>       ;    ESI = Path Array Address
3129                              <1>       ;    EDI = Current Directory String Buffer
3130                              <1>       ;    AH = Current Directory Level
3131                              <1>       ; OUTPUT => EAX, EBX, ESI will be changed
3132                              <1>       ;    EDI will be same with input
3133                              <1>       ;    ECX = Current Directory String Length
3134                              <1>
3135 0000AE48 57                  <1>       push    edi
3136 0000AE49 80FC00              <1>       cmp     ah, 0
3137 0000AE4C 7652                <1>       jna    short pass_write_path
3138 0000AE4E 83C610              <1>       add    esi, 16
3139 0000AE51 89F3                <1>       mov    ebx, esi
3140                              <1> loc_write_path:
3141 0000AE53 B908000000          <1>       mov    ecx, 8
3142                              <1> path_write_dirname1:
3143 0000AE58 AC                  <1>       lodsb
3144 0000AE59 3C20                <1>       cmp    al, 20h
3145 0000AE5B 7612                <1>       jna    short pass_write_dirname1
3146 0000AE5D AA                  <1>       stosb
3147 0000AE5E 81FF[148A0100]      <1>       cmp    edi, End_Of_Current_Dir_Str
3148 0000AE64 733A                <1>       jnb    short pass_write_path
3149 0000AE66 E2F0                <1>       loop   path_write_dirname1
3150 0000AE68 803E20              <1>       cmp    byte [esi], 20h
3151 0000AE6B 7624                <1>       jna    short pass_write_dirname2
3152 0000AE6D EB0A                <1>       jmp     short loc_put_dot_cont_ext
3153                              <1> pass_write_dirname1:
3154 0000AE6F 89DE                <1>       mov    esi, ebx
3155 0000AE71 83C608              <1>       add    esi, 8
3156 0000AE74 803E20              <1>       cmp    byte [esi], 20h
3157 0000AE77 7618                <1>       jna    short pass_write_dirname2
3158                              <1> loc_put_dot_cont_ext:
3159 0000AE79 C6072E              <1>       mov    byte [edi], "."
3160                              <1>       ;mov   ecx, 3
3161 0000AE7C B103                <1>       mov    cl, 3
3162                              <1> loc_check_dir_name_ext:
3163 0000AE7E AC                  <1>       lodsb
3164 0000AE7F 47                  <1>       inc    edi
3165 0000AE80 3C20                <1>       cmp    al, 20h
3166 0000AE82 760D                <1>       jna    short pass_write_dirname2
3167 0000AE84 8807                <1>       mov    [edi], al
3168 0000AE86 81FF[148A0100]      <1>       cmp    edi, End_Of_Current_Dir_Str
3169 0000AE8C 7312                <1>       jnb    short pass_write_path
3170 0000AE8E E2EE                <1>       loop    loc_check_dir_name_ext
3171 0000AE90 47                  <1>       inc    edi
3172                              <1> pass_write_dirname2:
3173 0000AE91 FECC                <1>       dec    ah
3174 0000AE93 740B                <1>       jz      short pass_write_path
3175 0000AE95 83C310              <1>       add    ebx, 16
3176 0000AE98 89DE                <1>       mov    esi, ebx
3177 0000AE9A C6072F              <1>       mov    byte [edi],"/"
3178 0000AE9D 47                  <1>       inc    edi
3179 0000AE9E EBB3                <1>       jmp    short loc_write_path
3180                              <1> pass_write_path:
3181 0000AEA0 C60700              <1>       mov    byte [edi], 0
3182 0000AEA3 47                  <1>       inc    edi
3183 0000AEA4 89F9                <1>       mov    ecx, edi
3184 0000AEA6 5F                  <1>       pop    edi
3185 0000AEA7 29F9                <1>       sub    ecx, edi
3186                              <1>       ; ECX = Current Directory String Length
3187 0000AEA9 C3                  <1>       retn
3188                              <1>
3189                              <1> get_current_directory:
3190                              <1>       ; 15/10/2016
3191                              <1>       ; 14/02/2016
3192                              <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
3193                              <1>       ; 27/03/2011
3194                              <1>       ;
3195                              <1>       ; INPUT-> ESI = Current Directory Buffer
3196                              <1>       ;         DL = TRDOS Logical Dos Drive Number + 1
3197                              <1>       ;              (0= Default/Current Drive)
3198                              <1>       ;
3199                              <1>       ;   Note: Required dir buffer length may be <= 92 bytes
3200                              <1>       ;         for TRDOS (7*12 name chars + 7 slash + 0)
3201                              <1>       ; OUTPUT ->  ESI = Current Directory Buffer
3202                              <1>       ;            EAX, EBX, ECX, EDX, EDI will be changed
3203                              <1>       ;            CX/CL = Current Directory String Length
3204                              <1>       ;           DL = Drive Number (0 based)
3205                              <1>       ;             (If input is 0, output is current drv number)
3206                              <1>       ;             DH = same with input
3207                              <1>       ;   cf = 0 -> AL = 0
3208                              <1>       ;   cf = 1 -> error code in AL
3209                              <1>
3210                              <1> loc_get_current_drive_0:
3211 0000AEAA 80FA00              <1>       cmp    dl, 0
3212 0000AEAD 7708                <1>       ja     short loc_get_current_drive_1
3213 0000AEAF 8A15[B6890100]      <1>       mov    dl, [Current_Drv]
3214 0000AEB5 EB17                <1>       jmp    short loc_get_current_drive_2
3215                              <1> loc_get_current_drive_1:
3216 0000AEB7 FECA                <1>       dec    dl
3217 0000AEB9 3A15[823F0100]      <1>       cmp    dl, [Last_DOS_DiskNo]
3218 0000AEBF 760D                <1>       jna    short loc_get_current_drive_2
3219 0000AEC1 B80F000000          <1>       mov    eax, 0Fh ; Invalid drive (Drive not ready!)
3220 0000AEC6 F5                  <1>       cmc    ; stc
```

```
3221 0000AEC7 C3                <1>       retn
3222                            <1>
3223                            <1> loc_get_current_drive_not_ready_retn:
3224 0000AEC8 5E                <1>       pop    esi
3225                            <1>       ;mov   eax, 15
3226 0000AEC9 66B80F00          <1>       mov    ax, 15 ; Drive not ready
3227 0000AECD C3                <1>       retn
3228                            <1>
3229                            <1> loc_get_current_drive_2:
3230 0000AECE 31C0              <1>       xor    eax, eax
3231 0000AED0 88D4              <1>       mov    ah, dl
3232 0000AED2 56                <1>       push   esi
3233 0000AED3 BE00010900        <1>       mov    esi, Logical_DOSDisks
3234 0000AED8 01C6              <1>       add    esi, eax
3235 0000AEDA 8A06              <1>       mov    al, [esi+LD_Name]
3236 0000AEDC 3C41              <1>       cmp    al, 'A'
3237 0000AEDE 72E8              <1>       jb     short loc_get_current_drive_not_ready_retn
3238                            <1>
3239 0000AEE0 8A667F            <1>       mov    ah, [esi+LD_CDirLevel]
3240 0000AEE3 08E4              <1>       or     ah, ah
3241 0000AEE5 7506              <1>       jnz    short loc_get_current_drive_3
3242                            <1>
3243                            <1>       ;xor   ah, ah ; mov ah, 0
3244 0000AEE7 8826              <1>       mov    [esi], ah
3245 0000AEE9 31C9              <1>       xor    ecx, ecx
3246 0000AEEB EB1C              <1>       jmp    short loc_get_current_drive_4
3247                            <1>
3248                            <1> loc_get_current_drive_3:
3249 0000AEED BF[13910100]      <1>         mov     edi, PATH_Array
3250 0000AEF2 57                <1>       push   edi
3251 0000AEF3 81C680000000      <1>       add    esi, LD_CurrentDirectory
3252 0000AEF9 B920000000        <1>       mov    ecx, 32
3253 0000AEFE F3A5              <1>       rep    movsd
3254 0000AF00 5E                <1>       pop    esi ; Path Array Address
3255 0000AF01 5F                <1>       pop    edi ; pushed esi (current dir buffer offset)
3256                            <1>       ;
3257 0000AF02 E841FFFFFF        <1>       call   set_current_directory_string
3258 0000AF07 89FE              <1>       mov    esi, edi
3259                            <1>
3260                            <1> loc_get_current_drive_4:
3261 0000AF09 30C0              <1>       xor    al, al
3262 0000AF0B C3                <1>       retn
3263                            <1>
3264                            <1> change_current_directory:
3265                            <1>       ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
3266                            <1>       ; 19/02/2016
3267                            <1>       ; 11/02/2016
3268                            <1>       ; 10/02/2016
3269                            <1>       ; 08/02/2016
3270                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3271                            <1>       ; 18/09/2011 (DIR.ASM, 09/10/2011)
3272                            <1>       ; 04/10/2009
3273                            <1>       ; 2005
3274                            <1>       ; INPUT ->
3275                            <1>       ;      ESI = Directory string
3276                            <1>       ;      ah = CD command (CDh = save current dir string)
3277                            <1>       ; OUTPUT ->
3278                            <1>       ;      EDI = DOS Drive Description Table
3279                            <1>       ;      cf = 1 -> error
3280                            <1>       ;         EAX = Error code
3281                            <1>       ;      cf = 0 -> successful
3282                            <1>       ;         ESI = PATH_Array
3283                            <1>       ;         EAX = Current Directory First Cluster
3284                            <1>       ;
3285                            <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
3286                            <1>
3287 0000AF0C 8825[A1910100]    <1>       mov    [CD_COMMAND], ah
3288 0000AF12 803E2F            <1>       cmp    byte [esi], '/'
3289 0000AF15 7505              <1>       jne    short loc_ccd_cdir_level
3290 0000AF17 46                <1>       inc    esi
3291 0000AF18 30C0              <1>       xor    al, al
3292 0000AF1A EB05              <1>       jmp    short loc_ccd_parse_path_name
3293                            <1> loc_ccd_cdir_level:
3294 0000AF1C A0[B4890100]      <1>       mov    al, [Current_Dir_Level]
3295                            <1> loc_ccd_parse_path_name:
3296 0000AF21 88C4              <1>       mov    ah, al
3297 0000AF23 BF[13910100]      <1>       mov    edi, PATH_Array
3298                            <1>
3299                            <1> ; Reset directory levels > cdir level
3300                            <1>       ; is this required !?
3301                            <1>       ;
3302                            <1>       ; Relations:
3303                            <1>       ; MAINPROG.ASM (pass_ccdrv_reset_cdir_FAT_fcluster)
3304                            <1>       ; proc_parse_dir_name,
3305                            <1>       ; proc_change_current_directory (this procedure)
3306                            <1>       ; proc_change_prompt_dir_string
3307                            <1>
3308 0000AF28 0FB6C8            <1>       movzx  ecx, al
3309 0000AF2B FEC1              <1>       inc    cl
3310 0000AF2D C0E104            <1>       shl    cl, 4
3311 0000AF30 01CF              <1>       add    edi, ecx
3312 0000AF32 B107              <1>       mov    cl, 7
3313 0000AF34 28C1              <1>       sub    cl, al
3314 0000AF36 C0E102            <1>       shl    cl, 2
3315 0000AF39 89C3              <1>       mov    ebx, eax
3316 0000AF3B 31C0              <1>       xor    eax, eax ; 0
3317 0000AF3D F3AB              <1>       rep    stosd
3318 0000AF3F 89D8              <1>       mov    eax, ebx
3319                            <1>
3320 0000AF41 BF[13910100]      <1>       mov    edi, PATH_Array
3321                            <1>
3322 0000AF46 803E20            <1>       cmp    byte [esi], 20h
3323 0000AF49 F5                <1>       cmc
3324 0000AF4A 7305              <1>       jnc    short pass_ccd_parse_dir_name
3325                            <1>
```

```
                                <1>             ; ESI = Path name
                                <1>             ; AL = CCD_Level
0000AF4C E871010000             <1>         call    parse_dir_name
                                <1>             ; AL = CCD_Level
                                <1>             ; AH = Last_Dir_Level
                                <1>             ; (EDI = PATH_Array)
                                <1>
                                <1> pass_ccd_parse_dir_name:
0000AF51 9C                     <1>         pushf
                                <1>
                                <1>         ;mov   [CCD_Level], al
                                <1>           ;mov [Last_Dir_Level], ah
0000AF52 66A3[97910100]         <1>         mov    [CCD_Level], ax
                                <1>
0000AF58 31DB                   <1>         xor    ebx, ebx
0000AF5A 8A3D[B6890100]         <1>         mov    bh, [Current_Drv]
0000AF60 BE00010900             <1>         mov    esi, Logical_DOSDisks
0000AF65 01DE                   <1>         add    esi, ebx
                                <1>
0000AF67 9D                     <1>         popf
0000AF68 720A                   <1>         jc     short loc_ccd_bad_path_name_retn
                                <1>
0000AF6A 8935[93910100]         <1>         mov    [CCD_DriveDT], esi
                                <1>
0000AF70 3C07                   <1>         cmp    al, 7
0000AF72 7209                   <1>         jb     short loc_ccd_load_child_dir
                                <1>
                                <1> loc_ccd_bad_path_name_retn:
0000AF74 87F7                   <1>         xchg   esi, edi
0000AF76 B813000000             <1>         mov    eax, 19 ; Bad directory/path name
0000AF7B F9                     <1>         stc
                                <1> loc_ccd_retn_p:
0000AF7C C3                     <1>         retn
                                <1>
                                <1> loc_ccd_load_child_dir:
                                <1>         ; AL = CCD_Level
0000AF7D 08C0                   <1>         or     al, al
0000AF7F 7467                   <1>         jz     short loc_ccd_load_root_dir
                                <1>
0000AF81 6689C1                 <1>         mov    cx, ax
0000AF84 C0E004                 <1>         shl    al, 4
0000AF87 0FB6F0                 <1>         movzx  esi, al
0000AF8A 01FE                   <1>         add    esi, edi  ; offset PATH_Array
                                <1>
0000AF8C 8B460C                 <1>         mov    eax, [esi+12]
0000AF8F 38E9                   <1>         cmp    cl, ch
0000AF91 0F84F9000000           <1>           je      loc_ccd_load_sub_directory
0000AF97 A3[B0890100]           <1>         mov    [Current_Dir_FCluster], eax
                                <1>
                                <1> loc_ccd_load_child_dir_next:
0000AF9C 83C610                 <1>         add    esi, 16 ; DOS DirEntry Format FileName Address
                                <1>
                                <1>         ; Directory attribute : 10h
0000AF9F B010                   <1>         mov    al, 00010000b ; 10h (Attrib AND mask)
                                <1>         ;mov   ah, 11001000b ; C8h
                                <1>         ; Volume name attribute: 8h
0000AFA1 B408                   <1>         mov    ah, 00001000b ; 08h (Attrib NAND, AND --> zero mask)
                                <1>
                                <1>         ;xor   cx, cx
0000AFA3 31C9                   <1>         xor    ecx, ecx ; 02/03/2021
0000AFA5 E8B5010000             <1>         call   locate_current_dir_file
0000AFAA 7353                   <1>         jnc    short loc_ccd_set_dir_cluster_ptr
                                <1>
                                <1>          ; 19/02/2016
                                <1>         ;mov   edi, [CCD_DriveDT]
0000AFAC 8A25[97910100]         <1>         mov    ah, [CCD_Level]
0000AFB2 803D[A1910100]CD       <1>         cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
0000AFB9 7509                   <1>         jne    short loc_ccd_load_child_dir_err
                                <1>         ; It is better to save recent successful part
                                <1>         ; of the (requested) path as current directory.
                                <1>         ; (Otherwise the path would be reset to back
                                <1>         ; on the next 'CD' command.)
0000AFBB 88E1                   <1>         mov    cl, ah
0000AFBD 50                     <1>         push   eax
0000AFBE E8E3000000             <1>         call   loc_ccd_save_current_dir
0000AFC3 58                     <1>         pop    eax
                                <1> loc_ccd_load_child_dir_err:
0000AFC4 3C03                   <1>         cmp    al, 3  ; AL = 2 => File not found error
0000AFC6 7202                   <1>         jb     short loc_ccd_path_not_found_retn
0000AFC8 F9                     <1>         stc
0000AFC9 C3                     <1>         retn
                                <1>
                                <1> loc_ccd_path_not_found_retn:
0000AFCA B003                   <1>         mov    al, 3  ; Path not found
0000AFCC C3                     <1>         retn
                                <1>
                                <1> loc_ccd_load_FAT_root_dir:
0000AFCD 803D[B5890100]02       <1>         cmp    byte [Current_FATType], 2
0000AFD4 776B                   <1>         ja     short loc_ccd_load_FAT32_root_dir
                                <1>
                                <1>         ;mov   esi, [CCD_DriveDT]
                                <1>         ;push  esi
0000AFD6 E8B61D0000             <1>         call   load_FAT_root_directory
                                <1>         ;pop   edi ; Dos Drv Description Table
                                <1>
0000AFDB 89F7                   <1>         mov    edi, esi
0000AFDD BE[13910100]           <1>         mov    esi, PATH_Array
0000AFE2 7298                   <1>         jc     short loc_ccd_retn_p
                                <1>
0000AFE4 31C0                   <1>         xor    eax, eax
0000AFE6 EB78                   <1>           jmp  short loc_ccd_set_cdfc
                                <1>
                                <1> loc_ccd_load_root_dir:
0000AFE8 803D[B5890100]01       <1>         cmp    byte [Current_FATType], 1
0000AFEF 73DC                   <1>         jnb    short loc_ccd_load_FAT_root_dir
```

```
                                <1>
                                <1> loc_ccd_load_FS_root_dir:
0000AFF1 E8621E0000             <1>         call    load_FS_root_directory
0000AFF6 EB5C                   <1>         jmp     short pass_ccd_load_FAT_sub_directory
                                <1>
                                <1> loc_ccd_load_FS_sub_directory_next:
0000AFF8 E85C1E0000             <1>         call    load_FS_sub_directory
0000AFFD EB1F                   <1>         jmp     short pass_ccd_set_dir_cluster_ptr
                                <1>
                                <1> loc_ccd_set_dir_cluster_ptr:
                                <1>         ; EDI = Directory Entry
0000AFFF 668B4714               <1>         mov     ax, [edi+20] ; First Cluster High Word
0000B003 C1E010                 <1>         shl     eax, 16
0000B006 668B471A               <1>         mov     ax, [edi+26] ; First Cluster Low Word
                                <1>
0000B00A 8B35[93910100]         <1>         mov     esi, [CCD_DriveDT]
0000B010 803D[B5890100]01       <1>         cmp     byte [Current_FATType], 1
0000B017 72DF                   <1>         jb      short loc_ccd_load_FS_sub_directory_next
                                <1>         ;push   esi
0000B019 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
0000B01E BE[13910100]           <1>         mov     esi, PATH_Array
0000B023 7264                   <1>         jc      short loc_ccd_retn_c
                                <1>
0000B025 A1[E1900100]           <1>         mov     eax, [DirBuff_Cluster]
                                <1>
0000B02A FE05[97910100]         <1>         inc     byte [CCD_Level]
0000B030 0FB61D[97910100]       <1>         movzx   ebx, byte [CCD_Level]
0000B037 C0E304                 <1>         shl     bl, 4 ; * 16 (<= 128)
0000B03A 01DE                   <1>         add     esi, ebx ; 19/02/2016
0000B03C 89460C                 <1>         mov     [esi+12], eax
0000B03F EB1F                   <1>         jmp     short loc_ccd_set_cdfc
                                <1>
                                <1> loc_ccd_load_FAT32_root_dir:
0000B041 BE[13910100]           <1>         mov     esi, PATH_Array
0000B046 8B460C                 <1>         mov     eax, [esi+12]
0000B049 8B35[93910100]         <1>         mov     esi, [CCD_DriveDT]
                                <1>
                                <1> loc_ccd_load_FAT_sub_directory:
                                <1>         ;push   esi
0000B04F 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
0000B054 BE[13910100]           <1>         mov     esi, PATH_Array
0000B059 722E                   <1>         jc      short loc_ccd_retn_c
                                <1>
0000B05B A1[E1900100]           <1>         mov     eax, [DirBuff_Cluster]
                                <1>
                                <1> loc_ccd_set_cdfc:
0000B060 8A0D[97910100]         <1>         mov     cl, [CCD_Level]
0000B066 880D[B4890100]         <1>         mov     [Current_Dir_Level], cl
0000B06C A3[B0890100]           <1>         mov     [Current_Dir_FCluster], eax
                                <1>
0000B071 8A2D[98910100]         <1>         mov     ch, [Last_Dir_Level]
0000B077 38E9                   <1>         cmp     cl, ch
0000B079 0F821DFFFFFF           <1>         jb      loc_ccd_load_child_dir_next
                                <1>
0000B07F 803D[A1910100]CD       <1>         cmp     byte [CD_COMMAND], 0CDh ;'CD' command or another
0000B086 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>
0000B088 F8                     <1>         clc
                                <1>
                                <1> loc_ccd_retn_c:
0000B089 8B3D[93910100]         <1>         mov     edi, [CCD_DriveDT]
0000B08F C3                     <1>         retn
                                <1>
                                <1> loc_ccd_load_sub_directory:
0000B090 8B35[93910100]         <1>         mov     esi, [CCD_DriveDT]
0000B096 803D[B5890100]01       <1>         cmp     byte [Current_FATType], 1
0000B09D 73B0                   <1>         jnb     short loc_ccd_load_FAT_sub_directory
0000B09F E8B51D0000             <1>         call    load_FS_sub_directory
0000B0A4 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>         ;
0000B0A6 BE[13910100]           <1>         mov     esi, PATH_Array ; 19/02/2016
0000B0AB 8B3D[93910100]         <1>         mov     edi, [CCD_DriveDT]
0000B0B1 57                     <1>         push    edi
                                <1>           add     edi, LD_CDirLevel
0000B0B2 83C77F                 <1>         mov     [edi], cl
0000B0B5 880F                   <1>         inc     edi ; LD_CurrentDirectory
0000B0B7 47                     <1>         push    esi
0000B0B8 56                     <1>         ;;mov   ecx, 32  ; always < 65536 (in this procedure)
                                <1>         mov     cx, 32
0000B0B9 66B92000               <1>         ; 02/03/2021
                                <1>         ;mov    ecx, 32
0000B0BD F3A5                   <1>         rep     movsd
                                <1>         ; Current directory has been saved to
                                <1>         ; the DOS drive description table, cdir area !
0000B0BF 5E                     <1>         pop     esi  ; PATH_Array
0000B0C0 5F                     <1>         pop     edi  ; Dos Drv Description Table
                                <1>
0000B0C1 C3                     <1>         retn
                                <1>
```

```
                                <1> parse_dir_name:
                                <1>       ; 11/02/2016
                                <1>       ; 10/02/2016
                                <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 18/09/2011
                                <1>       ; 17/10/2009
                                <1>       ; INPUT ->
                                <1>       ;      ESI = ASCIIZ Directory String Address
                                <1>       ;      AL = Current Directory Level
                                <1>       ;      EDI = Destination Adress
                                <1>       ;           (8 levels, each one 12+4 byte)
                                <1>       ; OUTPUT ->
                                <1>       ;      EDI = Dir Entry Formatted Array
                                <1>       ;           with zero cluster pointer at the last level
                                <1>       ;      AH = Last Dir Level
                                <1>       ;      AL = Current Dir Level
                                <1>       ;
                                <1>       ; (esi, ebx, ecx will be changed)
                                <1>
                                <1>       ;mov   [PATH_Array_Ptr], edi
0000B0C2 88C4                   <1>       mov    ah, al
0000B0C4 66A3[38920100]         <1>       mov    [PATH_CDLevel], ax
                                <1> repeat_ppdn_check_slash:
0000B0CA AC                     <1>       lodsb
0000B0CB 3C2F                   <1>       cmp    al, '/'
0000B0CD 74FB                   <1>       je     short repeat_ppdn_check_slash
0000B0CF 3C21                   <1>       cmp    al, 21h
0000B0D1 7219                   <1>       jb     short loc_ppdn_retn
0000B0D3 57                     <1>       push   edi
                                <1> loc_ppdn_get_dir_name:
0000B0D4 B90C000000             <1>       mov    ecx, 12
0000B0D9 BF[3A920100]           <1>       mov    edi, Dir_File_Name
                                <1> repeat_ppdn_get_dir_name:
0000B0DE AA                     <1>       stosb
0000B0DF AC                     <1>       lodsb
0000B0E0 3C2F                   <1>       cmp    al, '/'
0000B0E2 740A                   <1>       je     short loc_check_level_dot_conv_dir_name
0000B0E4 3C20                   <1>       cmp    al, 20h
0000B0E6 7605                   <1>       jna    short loc_ppdn_end_of_path_scan
0000B0E8 E2F4                   <1>       loop   repeat_ppdn_get_dir_name
0000B0EA 5F                     <1>       pop    edi
0000B0EB F9                     <1>       stc
                                <1> loc_ppdn_retn:
0000B0EC C3                     <1>       retn
                                <1>
                                <1> loc_ppdn_end_of_path_scan:
0000B0ED 4E                     <1>       dec    esi
                                <1> loc_check_level_dot_conv_dir_name:
0000B0EE 31C0                   <1>       xor    eax, eax
0000B0F0 AA                     <1>       stosb
0000B0F1 89F3                   <1>       mov    ebx, esi
0000B0F3 BE[3A920100]           <1>       mov    esi, Dir_File_Name
0000B0F8 AC                     <1>       lodsb
                                <1> repeat_ppdn_name_check_dot:
0000B0F9 3C2E                   <1>       cmp    al, '.'
0000B0FB 7509                   <1>       jne    short loc_ppdn_convert_sub_dir_name
                                <1> repeat_ppdn_name_dot_dot:
0000B0FD AC                     <1>       lodsb
0000B0FE 3C2E                   <1>       cmp    al, '.'
0000B100 743E                   <1>       je     short loc_ppdn_dot_dot
0000B102 3C21                   <1>       cmp    al, 21h
0000B104 7226                   <1>       jb     short pass_ppdn_convert_sub_dir_name
                                <1> loc_ppdn_convert_sub_dir_name:
0000B106 8A25[39920100]         <1>       mov    ah, [PATH_Level]
0000B10C 80FC07                 <1>       cmp    ah, 7
0000B10F 731B                   <1>       jnb    short pass_ppdn_convert_sub_dir_name
0000B111 FEC4                   <1>       inc    ah
0000B113 8825[39920100]         <1>       mov    [PATH_Level], ah
0000B119 BE[3A920100]           <1>       mov    esi, Dir_File_Name
                                <1>       ;mov   edi, [PATH_Array_Ptr]
0000B11E B010                   <1>       mov    al, 16
0000B120 F6E4                   <1>       mul    ah
0000B122 8B3C24                 <1>       mov    edi, [esp]
                                <1>       ;push  edi
0000B125 01C7                   <1>       add    edi, eax
0000B127 E82B030000             <1>       call   convert_file_name
                                <1>       ;pop   edi
                                <1> pass_ppdn_convert_sub_dir_name:
0000B12C 89DE                   <1>       mov    esi, ebx
                                <1> repeat_ppdn_check_last_slash:
0000B12E AC                     <1>       lodsb
0000B12F 3C2F                   <1>       cmp    al, '/'
0000B131 74FB                   <1>       je     short repeat_ppdn_check_last_slash
0000B133 3C21                   <1>       cmp    al, 21h
0000B135 739D                   <1>       jnb    short loc_ppdn_get_dir_name
                                <1> end_of_parse_dir_name:
0000B137 5F                     <1>       pop    edi
0000B138 F5                     <1>       cmc
                                <1>       ;mov   al, [PATH_CDLevel]
                                <1>       ;mov   ah, [PATH_Level]
0000B139 66A1[38920100]         <1>       mov    ax, [PATH_CDLevel]
0000B13F C3                     <1>       retn
                                <1>
                                <1> loc_ppdn_dot_dot:
0000B140 AC                     <1>       lodsb
0000B141 3C21                   <1>       cmp    al, 21h
0000B143 73F2                   <1>       jnb    short end_of_parse_dir_name
                                <1> loc_ppdn_dot_dot_prev_level:
0000B145 66A1[38920100]         <1>       mov    ax, [PATH_CDLevel]
0000B14B 80EC01                 <1>       sub    ah, 1
0000B14E 80D400                 <1>       adc    ah, 0
0000B151 38E0                   <1>       cmp    al, ah
0000B153 7602                   <1>       jna    short pass_ppdn_set_al_to_ah
0000B155 88E0                   <1>       mov    al, ah
                                <1> pass_ppdn_set_al_to_ah:
```

```
3641 0000B157 66A3[38920100]      <1>       mov    [PATH_CDLevel], ax
3642 0000B15D EBCD                <1>       jmp    short pass_ppdn_convert_sub_dir_name
3643                              <1>
3644                              <1> locate_current_dir_file:
3645                              <1>       ; 20/11/2017
3646                              <1>       ; 14/02/2016
3647                              <1>       ; 13/02/2016
3648                              <1>       ; 10/02/2016
3649                              <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3650                              <1>       ; 14/08/2010
3651                              <1>       ; 19/09/2009
3652                              <1>         ; 2005
3653                              <1>       ; INPUT ->
3654                              <1>       ;     ESI = DOS DirEntry Format FileName Address
3655                              <1>       ;     AL = Attributes Mask
3656                              <1>       ;     (<AL AND EntryAttrib> must be equal to AL)
3657                              <1>       ;     AH = Negative Attributes Mask (If AH>0)
3658                              <1>       ;     (<AH AND EntryAttrib> must be ZERO)
3659                              <1>       ;     CH > 0 Find First Free Dir Entry or Deleted Entry
3660                              <1>       ;     CL = 0 -> Return the First Free Dir Entry
3661                              <1>       ;     CL = E5h -> Return the 1st deleted entry
3662                              <1>       ;     CL = FFh -> Return the 1st deleted or free entry
3663                              <1>       ;     CL > 0 and CL <> E5h and CL <> FFh -> Return the first
3664                              <1>       ;          proper entry (which fits with Atributes Masks)
3665                              <1>       ;     CX = 0 Find Valid File/Directory/VolumeName
3666                              <1>       ;     ? = Any One Char
3667                              <1>       ;     * = Every Chars
3668                              <1>       ; OUTPUT ->
3669                              <1>       ;     EDI = Directory Entry Address (in Directory Buffer)
3670                              <1>       ;     ESI = DOS DirEntry Format FileName Address
3671                              <1>       ;     CF = 0 -> No Error, Proper Entry,
3672                              <1>       ;     DL = Attributes
3673                              <1>       ;     DH = Previous Entry Attr (LongName Check)
3674                              <1>       ;     AL > 0 -> Ambiguous filename wildcard "?" used
3675                              <1>       ;     AH > 0 -> Ambiguous filename wildcard "*" used
3676                              <1>       ;     AX = 0 -> Filename full fits with directory entry
3677                              <1>       ;     CH = The 1st Name Char of Current Dir Entry
3678                              <1>       ;     CF = 1 -> Proper entry not found, Error Code in EAX/AL
3679                              <1>       ;     CL = 0 and CH = 0 -> Free Entry (End Of Dir)
3680                              <1>       ;     CL = 0 and CH = E5h -> Deleted Entry fits with filters
3681                              <1>       ;     CL > 0 -> Entry not found, CH invalid
3682                              <1>       ;     CF = 0 ->
3683                              <1>       ;     EBX = Current Directory Entry Index/Number (BX)
3684                              <1>
3685                              <1>       ;mov   word [DirBuff_EntryCounter], 0 ; Zero Based
3686                              <1>
3687 0000B15F 8935[9B910100]      <1>       mov    [CDLF_FNAddress], esi
3688 0000B165 66A3[99910100]      <1>       mov    [CDLF_AttributesMask], ax
3689 0000B16B 66890D[9F910100]    <1>       mov    [CDLF_DEType], cx
3690                              <1>
3691 0000B172 31DB                <1>       xor    ebx, ebx
3692 0000B174 881D[B0910100]      <1>       mov    [PreviousAttr], bl ; 0  ; 13/02/2016
3693                              <1>
3694 0000B17A 8A3D[B6890100]      <1>       mov    bh, [Current_Drv]
3695 0000B180 381D[DC900100]      <1>       cmp    byte [DirBuff_ValidData], bl ; 0
3696 0000B186 761D                <1>       jna    short loc_lcdf_reload_current_dir2
3697 0000B188 8A1D[DA900100]      <1>         mov     bl, [DirBuff_DRV]
3698 0000B18E 80EB41              <1>       sub    bl, 'A'
3699 0000B191 38DF                <1>       cmp    bh, bl
3700 0000B193 750E                <1>       jne    short loc_lcdf_reload_current_dir1
3701 0000B195 8B15[E1900100]      <1>       mov    edx, [DirBuff_Cluster]
3702 0000B19B 3B15[B0890100]      <1>       cmp    edx, [Current_Dir_FCluster]
3703 0000B1A1 7412                <1>       je     short loc_cdir_locatefile_search
3704                              <1>
3705                              <1> loc_lcdf_reload_current_dir1:
3706 0000B1A3 30DB                <1>       xor    bl, bl
3707                              <1> loc_lcdf_reload_current_dir2:
3708 0000B1A5 89DE                <1>       mov    esi, ebx
3709 0000B1A7 81C600010900        <1>         add     esi, Logical_DOSDisks
3710 0000B1AD E874000000          <1>       call   reload_current_directory
3711 0000B1B2 735D                <1>       jnc    short loc_locatefile_search_again
3712 0000B1B4 C3                  <1>       retn
3713                              <1>
3714                              <1> loc_cdir_locatefile_search:
3715 0000B1B5 31DB                <1>       xor    ebx, ebx
3716 0000B1B7 55                  <1>       push   ebp ; 20/11/2017
3717 0000B1B8 E8A6000000          <1>       call   find_directory_entry
3718 0000B1BD 5D                  <1>       pop    ebp ; 20/11/2017
3719 0000B1BE 7349                <1>       jnc    short loc_cdir_locate_file_retn
3720                              <1>
3721                              <1> loc_locatefile_check_stc_reason:
3722 0000B1C0 08ED                <1>       or     ch, ch
3723 0000B1C2 7444                <1>       jz     short loc_cdir_locate_file_stc_retn
3724                              <1>
3725                              <1> loc_locatefile_check_next_entryblock:
3726 0000B1C4 8A3D[B6890100]      <1>       mov    bh, [Current_Drv]
3727 0000B1CA 28DB                <1>       sub    bl, bl
3728 0000B1CC 0FB7F3              <1>       movzx  esi, bx
3729 0000B1CF 81C600010900        <1>         add     esi, Logical_DOSDisks
3730                              <1>
3731 0000B1D5 803D[B4890100]00    <1>       cmp    byte [Current_Dir_Level], 0
3732 0000B1DC 760A                <1>       jna    short loc_locatefile_check_FAT_type
3733                              <1>
3734 0000B1DE 803D[B5890100]01    <1>       cmp    byte [Current_FATType], 1
3735 0000B1E5 730A                <1>       jnb    short loc_locatefile_load_subdir_cluster
3736 0000B1E7 C3                  <1>       retn
3737                              <1>
3738                              <1> loc_locatefile_check_FAT_type:
3739 0000B1E8 803D[B5890100]03    <1>       cmp    byte [Current_FATType], 3
3740 0000B1EF 7218                <1>       jb     short loc_cdir_locate_file_retn
3741                              <1>
3742                              <1> loc_locatefile_load_subdir_cluster:
3743 0000B1F1 A1[E1900100]        <1>       mov    eax, [DirBuff_Cluster]
3744 0000B1F6 E83B1A0000          <1>       call   get_next_cluster
3745 0000B1FB 730D                <1>       jnc    short loc_locatefile_next_cluster
```

```
3746 0000B1FD 09C0              <1>       or    eax, eax
3747 0000B1FF 7507              <1>       jnz   short loc_locatefile_drive_not_ready_read_err
3748 0000B201 F9                <1>       stc
3749                            <1> loc_locatefile_file_notfound:
3750 0000B202 B802000000        <1>       mov   eax, 2 ; File/Directory/VolName not found
3751 0000B207 C3                <1>       retn
3752                            <1>
3753                            <1> loc_locatefile_drive_not_ready_read_err:
3754                            <1>       ;mov  eax, 17 ;Drive not ready or read error
3755                            <1> loc_cdir_locate_file_stc_retn:
3756 0000B208 F5                <1>       cmc ;stc
3757                            <1> loc_cdir_locate_file_retn:
3758 0000B209 C3                <1>       retn
3759                            <1>
3760                            <1> loc_locatefile_next_cluster:
3761 0000B20A E80D1C0000        <1>       call  load_FAT_sub_directory
3762                            <1>       ;jc   short loc_locatefile_drive_not_ready_read_err
3763 0000B20F 72F8              <1>       jc    short loc_cdir_locate_file_retn
3764                            <1>
3765                            <1> loc_locatefile_search_again:
3766 0000B211 8B35[9B910100]    <1>       mov   esi, [CDLF_FNAddress]
3767 0000B217 66A1[99910100]    <1>       mov   ax, [CDLF_AttributesMask]
3768 0000B21D 668B0D[9F910100]  <1>       mov   cx, [CDLF_DEType]
3769 0000B224 EB8F              <1>       jmp   short loc_cdir_locatefile_search
3770                            <1>
3771                            <1> reload_current_directory:
3772                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3773                            <1>       ; 13/06/2010
3774                            <1>       ; 22/09/2009
3775                            <1>         ;
3776                            <1>       ; INPUT ->
3777                            <1>       ;     ESI = Dos drive description table address
3778                            <1>
3779                            <1>       ;mov  al, [esi+LD_FATType]
3780 0000B226 A0[B5890100]      <1>       mov   al, [Current_FATType]
3781 0000B22B 3C02              <1>       cmp   al, 2
3782 0000B22D 7729              <1>       ja    short loc_reload_FAT_sub_directory
3783 0000B22F 8A25[B4890100]    <1>       mov   ah, [Current_Dir_Level]
3784 0000B235 08C0              <1>       or    al, al
3785 0000B237 740A              <1>       jz    short loc_reload_FS_directory
3786 0000B239 08E4              <1>       or    ah, ah
3787 0000B23B 751B              <1>       jnz   short loc_reload_FAT_sub_directory
3788                            <1> loc_reload_FAT_12_16_root_directory:
3789 0000B23D E84F1B0000        <1>       call  load_FAT_root_directory
3790 0000B242 C3                <1>       retn
3791                            <1> loc_reload_FS_directory:
3792 0000B243 20E4              <1>       and   ah, ah
3793 0000B245 7506              <1>       jnz   short loc_reload_FS_sub_directory
3794                            <1> loc_reload_FS_root_directory:
3795 0000B247 E80C1C0000        <1>       call  load_FS_root_directory
3796 0000B24C C3                <1>       retn
3797                            <1> loc_reload_FS_sub_directory:
3798 0000B24D A1[B0890100]      <1>       mov   eax, [Current_Dir_FCluster]
3799 0000B252 E8021C0000        <1>       call  load_FS_sub_directory
3800 0000B257 C3                <1>       retn
3801                            <1> loc_reload_FAT_sub_directory:
3802 0000B258 A1[B0890100]      <1>       mov   eax, [Current_Dir_FCluster]
3803 0000B25D E8BA1B0000        <1>       call  load_FAT_sub_directory
3804 0000B262 C3                <1>       retn
3805                            <1>
3806                            <1> find_directory_entry:
3807                            <1>       ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
3808                            <1>       ; 14/02/2016
3809                            <1>       ; 13/02/2016
3810                            <1>       ; 10/02/2016
3811                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3812                            <1>       ; 14/08/2010 (DIR.ASM, "proc_find_direntry")
3813                            <1>       ; 19/09/2009
3814                            <1>       ; 2005
3815                            <1>       ; INPUT ->
3816                            <1>       ;     ESI = Sub Dir or File Name Address
3817                            <1>       ;     AL = Attributes Mask
3818                            <1>       ;     (<AL AND EntryAttrib> must be equal to AL)
3819                            <1>       ;     AH = Negative Attributes Mask (If AH>0)
3820                            <1>       ;     (<AH AND EntryAttrib> must be ZERO)
3821                            <1>       ;     CH > 0 Find First Free Dir Entry or Deleted Entry
3822                            <1>       ;     CL = 0 -> Return the First Free Dir Entry
3823                            <1>       ;     CL = E5h -> Return the 1st deleted entry
3824                            <1>       ;     CL = FFh -> Return the 1st deleted or free entry
3825                            <1>       ;     CL > 0 and CL <> E5h and CL <> FFh -> Return the first
3826                            <1>       ;           proper entry (which fits with Atributes Masks)
3827                            <1>       ;     CX = 0 -> Find Valid File/Directory/VolumeName
3828                            <1>       ;     ? = Any One Char
3829                            <1>       ;     * = Every Chars
3830                            <1>       ;     EBX = Current Dir Entry (BX)
3831                            <1>       ;
3832                            <1>       ; OUTPUT ->
3833                            <1>       ;     EDI = Directory Entry Address (in DirectoryBuffer)
3834                            <1>       ;     ESI = Sub Dir or File Name Address
3835                            <1>       ;     CF = 0 -> No Error, Proper Entry,
3836                            <1>       ;     DL = Attributes
3837                            <1>       ;     DH = Previous Entry Attr (LongName Check)
3838                            <1>       ;     AL > 0 -> Ambiguous filename wildcard "?" used
3839                            <1>       ;     AH > 0 -> Ambiguous filename wildcard "*" used
3840                            <1>       ;     AX = 0 -> Filename full fits with directory entry
3841                            <1>       ;     EBX = CurrentDirEntry (BX)
3842                            <1>       ;     CH = The 1st Name Char of Current Dir Entry
3843                            <1>       ;     CF = 1 -> Proper entry not found, Error Code in AX/AL
3844                            <1>       ;     CL = 0 and CH = 0 -> Free Entry (End Of Dir)
3845                            <1>       ;     CL = 0 and CH = E5h -> Deleted Entry fits with filters
3846                            <1>       ;     CL > 0 -> Entry not found, CH invalid
3847                            <1>       ;
3848                            <1>       ; (EAX, EBX, ECX, EDX, EDI, EBP will be changed)
3849                            <1>
3850 0000B263 663B1D[DF900100]  <1>       cmp   bx, [DirBuff_LastEntry]
```

```
3851 0000B26A 0F8739010000        <1>         ja      loc_ffde_stc_retn_255
3852                              <1>
3853                              <1>       ;mov    [DirBuff_CurrentEntry], bx
3854                              <1>
3855 0000B270 BF00000800          <1>       mov    edi, Directory_Buffer
3856 0000B275 66A3[AC910100]      <1>       mov    [FDE_AttrMask], ax
3857                              <1>
3858 0000B27B 29C0                <1>       sub    eax, eax
3859                              <1>
3860                              <1>       ;;mov  [PreviousAttr], al ; 0 ;; 13/02/2016
3861 0000B27D 66A3[AE910100]      <1>       mov    [AmbiguousFileName], ax ; 0
3862                              <1>
3863 0000B283 6689D8              <1>       mov    ax, bx
3864 0000B286 66C1E005            <1>       shl    ax, 5 ; ; * 32 ; Directory entry size
3865 0000B28A 01C7                <1>       add     edi, eax
3866                              <1>
3867 0000B28C 08ED                <1>       or     ch, ch
3868 0000B28E 0F852D010000        <1>         jnz     loc_find_free_deleted_entry_0
3869                              <1>
3870 0000B294 08C9                <1>       or      cl, cl
3871 0000B296 0F850D010000        <1>         jnz     loc_ffde_stc_retn_255
3872                              <1>
3873                              <1> check_find_dir_entry:
3874 0000B29C 66A1[AC910100]      <1>       mov    ax, [FDE_AttrMask]
3875 0000B2A2 8A2F                <1>       mov    ch, [edi]
3876 0000B2A4 80FD00              <1>       cmp     ch, 0 ; Is it never used entry?
3877 0000B2A7 0F8600010000        <1>       jna    loc_find_direntry_stc_retn
3878 0000B2AD 56                  <1>       push   esi
3879 0000B2AE 8A570B              <1>       mov    dl, [edi+0Bh] ; File attributes
3880 0000B2B1 80FDE5              <1>       cmp    ch, 0E5h ; Is it a deleted file?
3881 0000B2B4 746D                <1>       je     short loc_find_dir_next_entry_prevdeleted
3882                              <1>
3883 0000B2B6 80FA0F              <1>       cmp     dl, 0Fh ; longname sub component check
3884 0000B2B9 7505                <1>       jne     short loc_check_attributes_mask
3885 0000B2BB E8EE010000          <1>       call   save_longname_sub_component
3886                              <1>
3887                              <1> loc_check_attributes_mask:
3888 0000B2C0 88C6                <1>       mov    dh, al
3889 0000B2C2 20D6                <1>       and    dh, dl
3890 0000B2C4 38F0                <1>       cmp    al, dh
3891 0000B2C6 0F85BA000000        <1>         jne     loc_find_dir_next_entry
3892 0000B2CC 20D4                <1>       and    ah, dl
3893 0000B2CE 0F85B2000000        <1>         jnz     loc_find_dir_next_entry
3894 0000B2D4 80FA0F              <1>       cmp    dl, 0Fh
3895 0000B2D7 751A                <1>       jne    short pass_direntry_attr_check
3896                              <1>
3897 0000B2D9 3C0F                <1>       cmp    al, 0Fh ; AL = 0Fh -> find long name
3898 0000B2DB 0F85A5000000        <1>         jne     loc_find_dir_next_entry
3899                              <1>
3900 0000B2E1 5E                  <1>       pop    esi
3901 0000B2E2 6631C0              <1>       xor    ax, ax
3902 0000B2E5 8A35[B0910100]      <1>       mov    dh, [PreviousAttr]
3903 0000B2EB 66891D[DD900100]    <1>       mov    [DirBuff_CurrentEntry], bx
3904 0000B2F2 C3                  <1>       retn
3905                              <1>
3906                              <1> pass_direntry_attr_check:
3907 0000B2F3 89FD                <1>       mov    ebp, edi ; 14/02/2016
3908 0000B2F5 B908000000          <1>       mov    ecx, 8
3909                              <1> loc_lodsb_find_dir:
3910 0000B2FA AC                  <1>       lodsb
3911 0000B2FB 3C2A                <1>       cmp    al, '*'
3912 0000B2FD 7508                <1>       jne    short pass_fde_ambiguous1_check
3913 0000B2FF FE05[AF910100]      <1>         inc     byte [AmbiguousFileName+1]
3914 0000B305 EB28                <1>       jmp    short loc_check_direntry_extension
3915                              <1>
3916                              <1> pass_fde_ambiguous1_check:
3917 0000B307 3C3F                <1>       cmp    al, '?'
3918 0000B309 750D                <1>       jne    short pass_fde_ambiguous2_check
3919 0000B30B FE05[AE910100]      <1>       inc    byte [AmbiguousFileName]
3920 0000B311 803F20              <1>       cmp    byte [edi], 20h
3921 0000B314 764E                <1>       jna    short loc_find_dir_next_entry_ebp
3922 0000B316 EB14                <1>       jmp    short loc_scasb_find_dir_inc_di
3923                              <1>
3924                              <1> pass_fde_ambiguous2_check:
3925 0000B318 3C20                <1>       cmp    al, 20h
3926 0000B31A 750C                <1>       jne    short loc_scasb_find_dir
3927 0000B31C 803F20              <1>       cmp    byte [edi], 20h
3928 0000B31F 7543                <1>       jne    short loc_find_dir_next_entry_ebp
3929 0000B321 EB0C                <1>       jmp    short loc_check_direntry_extension
3930                              <1>
3931                              <1> loc_find_dir_next_entry_prevdeleted:
3932 0000B323 80CA80              <1>       or     dl, 80h  ; Bit 7 -> deleted entry sign
3933 0000B326 EB5E                <1>       jmp    short loc_find_dir_next_entry
3934                              <1>
3935                              <1> loc_scasb_find_dir:
3936 0000B328 3A07                <1>       cmp    al, [edi]
3937 0000B32A 7538                <1>       jne    short loc_find_dir_next_entry_ebp
3938                              <1> loc_scasb_find_dir_inc_di:
3939 0000B32C 47                  <1>       inc    edi
3940 0000B32D E2CB                <1>       loop   loc_lodsb_find_dir
3941                              <1>
3942                              <1> loc_check_direntry_extension:
3943 0000B32F BE08000000          <1>       mov    esi, 8
3944 0000B334 89F7                <1>       mov    edi, esi ; 8
3945 0000B336 033424              <1>       add    esi, [esp] ; Sub Dir or File Name Address
3946 0000B339 01EF                <1>       add    edi, ebp
3947 0000B33B B103                <1>       mov    cl, 3
3948                              <1> loc_lodsb_find_dir_ext:
3949 0000B33D AC                  <1>       lodsb
3950 0000B33E 3C2A                <1>       cmp    al, '*'
3951 0000B340 7508                <1>       jne    short pass_fde_ambiguous3_check
3952 0000B342 FE05[AF910100]      <1>       inc    byte [AmbiguousFileName+1]
3953 0000B348 EB1E                <1>       jmp    short loc_find_dir_proper_direntry
3954                              <1>
3955                              <1> pass_fde_ambiguous3_check:
```

```
3956 0000B34A 3C3F              <1>       cmp    al, '?'
3957 0000B34C 750D              <1>       jne    short pass_fde_ambiguous4_check
3958 0000B34E FE05[AE910100]    <1>       inc    byte [AmbiguousFileName]
3959 0000B354 803F20            <1>       cmp    byte [edi], 20h
3960 0000B357 760B              <1>       jna    short loc_find_dir_next_entry_ebp
3961 0000B359 EB49              <1>       jmp    short loc_scasb_find_dir_ext_inc_di
3962                            <1>
3963                            <1> pass_fde_ambiguous4_check:
3964 0000B35B 3C20              <1>       cmp    al, 20h
3965 0000B35D 7541              <1>       jne    short loc_scasb_find_dir_ext
3966 0000B35F 803F20            <1>       cmp    byte [edi], 20h
3967 0000B362 7404              <1>       je     short loc_find_dir_proper_direntry
3968                            <1>
3969                            <1> loc_find_dir_next_entry_ebp:
3970 0000B364 89EF              <1>       mov    edi, ebp ; 14/02/2016
3971 0000B366 EB1E              <1>       jmp    short loc_find_dir_next_entry
3972                            <1>
3973                            <1> loc_find_dir_proper_direntry:
3974 0000B368 30C9              <1>       xor    cl, cl
3975                            <1> loc_find_dir_proper_direntry_1:
3976 0000B36A 5E                <1>       pop    esi
3977 0000B36B 89EF              <1>         mov     edi, ebp
3978 0000B36D 8A2F              <1>       mov    ch, [edi]
3979 0000B36F 8A570B            <1>       mov     dl, [edi+0Bh] ; Dir entry attributes
3980 0000B372 66A1[AE910100]    <1>       mov    ax, [AmbiguousFileName]
3981                            <1> loc_find_dir_proper_direntry_2:
3982 0000B378 8A35[B0910100]    <1>       mov     dh, [PreviousAttr]
3983 0000B37E 66891D[DD900100]  <1>       mov    [DirBuff_CurrentEntry], bx
3984 0000B385 C3                <1>       retn
3985                            <1>
3986                            <1> loc_find_dir_next_entry:
3987 0000B386 8815[B0910100]    <1>       mov    byte [PreviousAttr], dl ; LongName check
3988                            <1> loc_find_dir_next_entry_1:
3989 0000B38C 5E                <1>       pop    esi
3990 0000B38D 83C720            <1>       add    edi, 32
3991                            <1>       ;inc   word [DirBuff_EntryCounter]
3992 0000B390 6643              <1>       inc    bx
3993 0000B392 663B1D[DF900100]  <1>       cmp    bx, [DirBuff_LastEntry]
3994 0000B399 770E              <1>       ja     short loc_ffde_stc_retn_255
3995 0000B39B E9FCFEFFFF        <1>         jmp     check_find_dir_entry
3996                            <1>
3997                            <1> loc_scasb_find_dir_ext:
3998 0000B3A0 3A07              <1>       cmp    al, [edi]
3999 0000B3A2 75C0              <1>       jne    short loc_find_dir_next_entry_ebp
4000                            <1> loc_scasb_find_dir_ext_inc_di:
4001 0000B3A4 47                <1>       inc    edi
4002 0000B3A5 E296              <1>       loop    loc_lodsb_find_dir_ext
4003 0000B3A7 EBC1              <1>       jmp    short loc_find_dir_proper_direntry_1
4004                            <1>
4005                            <1> loc_ffde_stc_retn_255:
4006                            <1>       ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
4007                            <1>       ; (ECX must not be > 65535)
4008                            <1>       ; ((because 'loc_ccd_save_current_dir'
4009                            <1>       ;  sets CX to 32 for 'rep movsd'))
4010 0000B3A9 66B9FFFF          <1>       mov    cx, 0FFFFh
4011                            <1>       ;xor   ecx, ecx
4012                            <1>       ;dec   ecx ; 0FFFFFFFFh
4013                            <1>       ;xor   eax, eax
4014                            <1> loc_find_direntry_stc_retn:
4015                            <1> loc_check_ffde_retn_1:
4016                            <1>       ;mov   ax, 2
4017 0000B3AD B802000000        <1>       mov    eax, 2 ; File Not Found
4018 0000B3B2 8A35[B0910100]    <1>       mov    dh, [PreviousAttr]
4019 0000B3B8 66891D[DD900100]  <1>       mov    [DirBuff_CurrentEntry], bx
4020 0000B3BF F9                <1>       stc
4021 0000B3C0 C3                <1>       retn
4022                            <1>
4023                            <1> loc_find_free_deleted_entry_0:
4024 0000B3C1 66A1[AC910100]    <1>       mov    ax, [FDE_AttrMask]
4025 0000B3C7 8A2F              <1>       mov    ch, [edi]
4026 0000B3C9 8A570B            <1>       mov    dl, [edi+0Bh] ; File attributes
4027 0000B3CC 08C9              <1>       or     cl, cl
4028 0000B3CE 7407              <1>       jz     short loc_check_ffde_0_repeat
4029                            <1>       ;cmp   cl, 0E5h
4030                            <1>       ;je    short pass_loc_check_ffde_0_err
4031 0000B3D0 80F9FF            <1>       cmp    cl, 0FFh
4032 0000B3D3 7432              <1>       je     short loc_find_free_deleted_entry_1
4033 0000B3D5 EB4D              <1>       jmp    short pass_loc_check_ffde_0_err
4034                            <1>
4035                            <1> loc_check_ffde_0_repeat:
4036 0000B3D7 08ED              <1>       or     ch, ch
4037 0000B3D9 7511              <1>       jnz    short loc_check_ffde_0_next
4038                            <1>
4039                            <1> loc_check_ffde_retn_2:
4040 0000B3DB 6629C0            <1>       sub    ax, ax
4041 0000B3DE 8A35[B0910100]    <1>       mov    dh, [PreviousAttr]
4042 0000B3E4 66891D[DD900100]  <1>       mov    [DirBuff_CurrentEntry], bx
4043 0000B3EB C3                <1>       retn
4044                            <1>
4045                            <1> loc_check_ffde_0_next:
4046 0000B3EC 6643              <1>       inc    bx
4047 0000B3EE 83C720            <1>       add    edi, 32
4048                            <1>       ;inc   word [DirBuff_EntryCounter]
4049                            <1>
4050 0000B3F1 663B1D[DF900100]  <1>         cmp  bx, [DirBuff_LastEntry]
4051 0000B3F8 77AF              <1>       ja     short loc_ffde_stc_retn_255
4052 0000B3FA 8815[B0910100]    <1>       mov    [PreviousAttr], dl
4053 0000B400 8A2F              <1>       mov    ch, [edi]
4054 0000B402 8A570B            <1>       mov    dl, [edi+0Bh] ; file attributes
4055 0000B405 EBD0              <1>       jmp    short loc_check_ffde_0_repeat
4056                            <1>
4057                            <1> loc_find_free_deleted_entry_1:
4058 0000B407 28D2              <1>       sub    dl, dl
4059                            <1> loc_find_free_deleted_entry_2:
4060 0000B409 20ED              <1>       and    ch, ch
```

```
4061 0000B40B 74CE              <1>       jz    short loc_check_ffde_retn_2
4062 0000B40D 80FDE5            <1>       cmp   ch, 0E5h
4063 0000B410 74C9              <1>       je    short loc_check_ffde_retn_2
4064 0000B412 6643              <1>       inc   bx
4065 0000B414 83C720            <1>       add   edi, 32
4066 0000B417 663B1D[DF900100]  <1>       cmp   bx, [DirBuff_LastEntry]
4067 0000B41E 7789              <1>       ja    short loc_ffde_stc_retn_255
4068 0000B420 8A2F              <1>       mov   ch, [edi]
4069 0000B422 EBE5              <1>       jmp   short loc_find_free_deleted_entry_2
4070                            <1>
4071                            <1> pass_loc_check_ffde_0_err:
4072 0000B424 38CD              <1>       cmp   ch, cl
4073 0000B426 741F              <1>       je    short loc_check_ffde_attrib
4074                            <1>
4075 0000B428 6643              <1>       inc   bx
4076 0000B42A 83C720            <1>       add   edi, 32
4077 0000B42D 663B1D[DF900100]  <1>       cmp   bx, [DirBuff_LastEntry]
4078 0000B434 0F876FFFFFFF      <1>         ja      loc_ffde_stc_retn_255
4079 0000B43A 8815[B0910100]    <1>       mov   [PreviousAttr], dl
4080 0000B440 8A2F              <1>       mov   ch, [edi]
4081 0000B442 8A570B            <1>       mov   dl, [edi+0Bh]
4082 0000B445 EBDD              <1>       jmp   short pass_loc_check_ffde_0_err
4083                            <1>
4084                            <1> loc_check_ffde_attrib:
4085 0000B447 88C6              <1>       mov   dh, al
4086 0000B449 20D6              <1>       and   dh, dl
4087 0000B44B 38F0              <1>       cmp   al, dh
4088 0000B44D 759D              <1>       jne   short loc_check_ffde_0_next
4089 0000B44F 20D4              <1>       and   ah, dl
4090 0000B451 7599              <1>       jnz   short loc_check_ffde_0_next
4091 0000B453 30C9              <1>       xor   cl, cl
4092 0000B455 EB84              <1>         jmp     loc_check_ffde_retn_2
4093                            <1>
4094                            <1> convert_file_name:
4095                            <1>       ; 06/03/2016
4096                            <1>       ; 11/02/2016
4097                            <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
4098                            <1>       ; 06/10/2009
4099                            <1>       ; 2005
4100                            <1>       ;
4101                            <1>       ; INPUT  ->
4102                            <1>       ;     ESI = Dot File Name Location
4103                            <1>       ;     EDI = Dir Entry Format File Name Location
4104                            <1>       ; OUTPUT ->
4105                            <1>       ;     EDI = Dir Entry Format File Name Location
4106                            <1>       ;     ESI = Dot File Name Location (capitalized)
4107                            <1>       ;
4108                            <1>       ; (ECX, AL will be changed)
4109                            <1>
4110 0000B457 56                <1>       push  esi
4111 0000B458 57                <1>       push  edi
4112                            <1>
4113 0000B459 B90B000000        <1>       mov   ecx, 11
4114 0000B45E B020              <1>       mov   al, 20h
4115 0000B460 F3AA              <1>       rep   stosb
4116                            <1>
4117 0000B462 8B3C24            <1>       mov   edi, [esp]
4118                            <1>
4119 0000B465 B10C              <1>       mov   cl, 12 ; file name length (max.)
4120                            <1>       ; 06/03/2016
4121                            <1>       ; Directory entry name limit (11 bytes) check for
4122                            <1>       ; 'rename_directory_entry' procedure.
4123                            <1>       ; (EDI points to Directory Entry)
4124                            <1>       ; (If the file name would not contain a dot
4125                            <1>       ; and file name length would be 12, this would cause to
4126                            <1>       ; overwrite the attributes byte of the directory entry.)
4127                            <1>       ;
4128 0000B467 B50B              <1>       mov   ch, 11 ; directory entry's name length
4129                            <1> loc_check_first_dot:
4130 0000B469 8A06              <1>       mov   al, [esi]
4131 0000B46B 3C2E              <1>       cmp   al, 2Eh
4132 0000B46D 750C              <1>       jne   short pass_check_first_dot
4133 0000B46F 8807              <1>       mov   [edi], al
4134 0000B471 47                <1>       inc   edi
4135 0000B472 46                <1>       inc   esi
4136 0000B473 FEC9              <1>       dec   cl
4137 0000B475 75F2              <1>       jnz   short loc_check_first_dot
4138                            <1>       ;;(ecx <= 12)
4139                            <1>       ;;loop loc_check_first_dot
4140 0000B477 EB30              <1>       jmp   short stop_convert_file
4141                            <1>
4142                            <1> loc_get_fchar:
4143 0000B479 8A06              <1>       mov   al, [esi]
4144                            <1> pass_check_first_dot:
4145 0000B47B 3C61              <1>       cmp   al, 61h ; 'a'
4146 0000B47D 7208              <1>       jb    short pass_name_capitalize
4147 0000B47F 3C7A              <1>       cmp   al, 7Ah ; 'z'
4148 0000B481 7704              <1>       ja    short pass_name_capitalize
4149 0000B483 24DF              <1>       and   al, 0DFh
4150 0000B485 8806              <1>       mov   [esi], al
4151                            <1> pass_name_capitalize:
4152 0000B487 3C21              <1>       cmp   al, 21h
4153 0000B489 721E              <1>       jb    short stop_convert_file
4154 0000B48B 3C2E              <1>       cmp   al, 2Eh ; '.'
4155 0000B48D 750C              <1>       jne   short pass_dot_space
4156                            <1> add_dot_space:
4157 0000B48F 80F904            <1>       cmp   cl, 4
4158 0000B492 760E              <1>       jna   short inc_and_loop
4159 0000B494 47                <1>       inc   edi
4160 0000B495 FECD              <1>       dec   ch ; 06/03/2016
4161 0000B497 FEC9              <1>       dec   cl
4162 0000B499 EBF4              <1>       jmp   short add_dot_space
4163                            <1>
4164                            <1>       ;mov  al, 4
4165                            <1>       ;cmp  cl, al
```

```
                        <1>       ;jna   short inc_and_loop
                        <1>       ;sub   cl, al
                        <1>       ;add   edi, ecx
                        <1>       ;mov   cl, al
                        <1>       ;jmp   short inc_and_loop
                        <1>
                        <1> pass_dot_space:
0000B49B 8807           <1>       mov    [edi], al
                        <1> loc_after_double_dot:
                        <1>       ; 06/03/2016
0000B49D FECD           <1>       dec    ch ; count down for 11 bytes dir entry limit
0000B49F 740A           <1>       jz     short stop_convert_file_x
0000B4A1 47             <1>       inc    edi
                        <1> inc_and_loop:
0000B4A2 FEC9           <1>       dec    cl ; count down for 12 bytes filename limit
0000B4A4 7403           <1>       jz     short stop_convert_file
0000B4A6 46             <1>       inc    esi
                        <1>       ;;(ecx <= 12)
                        <1>       ;;loop loc_get_fchar
0000B4A7 EBD0           <1>       jmp    short loc_get_fchar
                        <1>
                        <1> stop_convert_file:
                        <1>       ; 06/03/2016
0000B4A9 30ED           <1>       xor    ch, ch
                        <1>       ; ECX < 256 ; 'find_first_file' -> xor cl, cl
                        <1> stop_convert_file_x:
0000B4AB 5F             <1>       pop    edi
0000B4AC 5E             <1>       pop    esi
0000B4AD C3             <1>       retn
                        <1>
                        <1> save_longname_sub_component:
                        <1>       ; 13/02/2016
                        <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 28/02/2010
                        <1>       ; 17/10/2009
                        <1>       ; INPUT ->
                        <1>       ;      EDI = Directory Entry
                        <1>       ;      // This procedure is called
                        <1>       ;      // from 'find_directory_entry' procedure.
                        <1>       ;      // If the last entry returns with
                        <1>       ;      // a non-zero LongnameFound value and
                        <1>       ;      // if LFN_CheckSum value is equal to
                        <1>       ;      // the next shortname checksum,
                        <1>       ;      // long name is valid.
                        <1>       ;      // If a longname is longer than 65 bytes,
                        <1>       ;      // it is invalid for trdos. (>45h)
                        <1>
0000B4AE 57             <1>       push   edi
0000B4AF 56             <1>       push   esi
                        <1>       ;push  ebx
                        <1>       ;push  ecx
                        <1>       ;push  edx
0000B4B0 50             <1>       push   eax
                        <1>
0000B4B1 29C9           <1>       sub    ecx, ecx
                        <1>       ;sub   eax, eax
0000B4B3 B11A           <1>       mov    cl, 26
                        <1>
0000B4B5 0FB607         <1>       movzx  eax, byte [edi] ; LDIR_Order
0000B4B8 3C41           <1>       cmp    al, 41h  ; 40h (last long entry sign) + 1
0000B4BA 722B           <1>       jb     short pass_pslnsc_last_long_entry
                        <1>
0000B4BC 88C4           <1>       mov    ah, al
0000B4BE 80EC40         <1>       sub    ah, 40h
0000B4C1 8825[B2910100] <1>       mov    [LFN_EntryLength], ah
                        <1>
0000B4C7 3C45           <1>       cmp    al, 45h  ; 40h (last long entry sign) + 5
                        <1>              ; Max 130 byte length is usable in TRDOS
                        <1> ; 26*5 = 130
0000B4C9 7753           <1>       ja     short loc_pslnsc_retn
                        <1>
0000B4CB 2407           <1>       and    al, 07h ; 0Fh
0000B4CD A2[B1910100]   <1>       mov    [LongNameFound], al
                        <1>
0000B4D2 FEC8           <1>       dec    al
                        <1>       ;mov   cl, 26
0000B4D4 F6E1           <1>       mul    cl
                        <1>
0000B4D6 89C6           <1>       mov    esi, eax
0000B4D8 01CE           <1>       add    esi, ecx
                        <1>              ; to make is an ASCIIZ string
                        <1>              ; with ax+26 bytes length
0000B4DA 81C6[B4910100] <1>       add    esi, LongFileName
0000B4E0 66C7060000     <1>       mov    word [esi], 0
0000B4E5 EB16           <1>       jmp    short loc_pslsc_move_ldir_name2
                        <1>
                        <1> pass_pslnsc_last_long_entry:
0000B4E7 3C04           <1>       cmp    al, 04h
0000B4E9 7733           <1>       ja     short loc_pslnsc_retn
0000B4EB FE0D[B1910100] <1>       dec    byte [LongNameFound]
0000B4F1 3A05[B1910100] <1>       cmp    al, [LongNameFound]
0000B4F7 7525           <1>       jne    short loc_pslnsc_retn
                        <1>
                        <1> loc_pslsc_move_ldir_name1:
0000B4F9 FEC8           <1>       dec    al
                        <1>       ;mov   cl, 26
0000B4FB F6E1           <1>       mul    cl
                        <1>
                        <1> loc_pslsc_move_ldir_name2:
0000B4FD 8A4F0D         <1>       mov    cl, [edi+0Dh] ; long name checksum
0000B500 880D[B3910100] <1>       mov    [LFN_CheckSum], cl
0000B506 89FE           <1>       mov    esi, edi ; LDIR_Order
0000B508 BF[B4910100]   <1>       mov    edi, LongFileName
0000B50D 01C7           <1>       add    edi, eax
0000B50F 46             <1>       inc    esi
```

```
4271 0000B510 B105                <1>       mov    cl, 5 ; chars 1 to 5
4272 0000B512 F366A5              <1>       rep    movsw
4273 0000B515 83C603              <1>       add    esi, 3
4274 0000B518 A5                  <1>       movsd  ; char 6 & 7
4275 0000B519 A5                  <1>       movsd  ; char 8 & 9
4276 0000B51A A5                  <1>       movsd  ; char 10 & 11
4277 0000B51B 46                  <1>       inc    esi
4278 0000B51C 46                  <1>       inc    esi
4279 0000B51D A5                  <1>       movsd   ; char 12 & 13
4280                              <1>
4281                              <1> loc_pslnsc_retn:
4282 0000B51E 58                  <1>       pop    eax
4283                              <1>       ;pop   edx
4284                              <1>       ;pop   ecx
4285                              <1>       ;pop   ebx
4286 0000B51F 5E                  <1>       pop    esi
4287 0000B520 5F                  <1>       pop    edi
4288                              <1>
4289 0000B521 C3                  <1>       retn
4290                              <1>
4291                              <1> parse_path_name:
4292                              <1>       ; 10/02/2016
4293                              <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
4294                              <1>       ; 10/009/2011 ('proc_parse_pathname')
4295                              <1>       ; 27/11/2009
4296                              <1>       ; 05/12/2004
4297                              <1>       ;
4298                              <1>       ; INPUT ->
4299                              <1>       ;      ESI = Beginning of ASCIIZ pathname string
4300                              <1>       ;       EDI = Destination Address
4301                              <1>       ;            (which is TR-DOS FindFile data buffer)
4302                              <1>       ; OUTPUT ->
4303                              <1>       ;      CF = 1 -> Error
4304                              <1>       ;           EAX = Error Code (AL)
4305                              <1>       ;
4306                              <1>       ; (Modified registers: eax, ecx, esi, edi)
4307                              <1>
4308                              <1>       ; Clear the pathname bytes in TR-DOS Findfile data buffer
4309 0000B522 57                  <1>       push   edi
4310 0000B523 B914000000          <1>       mov    ecx, 20  ; 80 bytes
4311 0000B528 31C0                <1>       xor    eax, eax
4312 0000B52A F3AB                <1>       rep    stosd
4313 0000B52C 5F                  <1>       pop    edi
4314                              <1>
4315 0000B52D 668B06              <1>       mov    ax, [esi]
4316 0000B530 80FC3A              <1>       cmp    ah, ':'
4317 0000B533 741C                <1>       je     short loc_ppn_change_drive
4318 0000B535 A0[B6890100]        <1>       mov    al, [Current_Drv]
4319 0000B53A EB33                <1>       jmp    short pass_ppn_change_drive
4320                              <1>
4321                              <1> pass_ppn_cdir:
4322 0000B53C 8B35[D6920100]      <1>       mov    esi, [First_Path_Pos]
4323 0000B542 AC                  <1>       lodsb
4324                              <1> loc_ppn_get_filename:
4325 0000B543 83C741              <1>       add    edi, 65 ; FindFile_Name location
4326                              <1>       ; TRDOS Filename length must not be more than 12 bytes
4327                              <1>       ;mov   ecx, 12
4328 0000B546 B10C                <1>       mov    cl, 12
4329                              <1> loc_ppn_get_fnchar_next:
4330 0000B548 AA                  <1>       stosb
4331 0000B549 AC                  <1>       lodsb
4332 0000B54A 3C21                <1>       cmp    al, 21h
4333 0000B54C 7274                <1>       jb     short loc_ppn_clc_return
4334 0000B54E E2F8                <1>         loop    loc_ppn_get_fnchar_next
4335                              <1> loc_ppn_return:
4336 0000B550 C3                  <1>       retn
4337                              <1>
4338                              <1> loc_ppn_change_drive:
4339 0000B551 24DF                <1>       and    al, 0DFh
4340 0000B553 2C41                <1>       sub    al, 'A'; A:
4341 0000B555 726F                <1>       jc     short loc_ppn_invalid_drive
4342 0000B557 3805[823F0100]      <1>       cmp    [Last_DOS_DiskNo], al
4343 0000B55D 7267                <1>       jb     short loc_ppn_invalid_drive
4344                              <1>
4345 0000B55F 46                  <1>       inc    esi
4346 0000B560 46                  <1>       inc    esi
4347 0000B561 8A26                <1>       mov    ah, [esi]
4348 0000B563 80FC21              <1>       cmp    ah, 21h
4349 0000B566 7307                <1>       jnb    short pass_ppn_change_drive
4350                              <1>
4351                              <1> loc_ppn_cmd_failed:
4352                              <1>       ; File or directory name is not existing
4353 0000B568 8807                <1>       mov    [edi], al ; Drv
4354 0000B56A 66B80100            <1>       mov    ax, 1 ; eax = 1
4355                              <1>       ; TR-DOS Error Code 01h = Bad Command Argument
4356                              <1>       ; MS-DOS Error Code 01h : Invalid Function Number
4357                              <1>       ;stc
4358                              <1>       ; (MainProg ErrMsg: "Bad command or file name!")
4359 0000B56E C3                  <1>       retn
4360                              <1>
4361                              <1> pass_ppn_change_drive:
4362 0000B56F 8935[D6920100]      <1>       mov    [First_Path_Pos], esi
4363 0000B575 C705[DA920100]0000- <1>       mov    dword [Last_Slash_Pos], 0
4363 0000B57D 0000                <1>
4364 0000B57F AA                  <1>       stosb
4365 0000B580 8A06                <1>       mov    al, [esi]
4366                              <1> loc_scan_ppn_dslash:
4367 0000B582 3C2F                <1>       cmp    al, '/'
4368 0000B584 7506                <1>       jne    short loc_scan_next_slash_pos
4369 0000B586 8935[DA920100]      <1>       mov    [Last_Slash_Pos], esi
4370                              <1> loc_scan_next_slash_pos:
4371 0000B58C 46                  <1>       inc    esi
4372 0000B58D 8A06                <1>       mov    al, [esi]
4373 0000B58F 3C20                <1>       cmp    al, 20h
4374 0000B591 77EF                <1>       ja     short loc_scan_ppn_dslash
```

```
4375 0000B593 833D[DA920100]00    <1>       cmp    dword [Last_Slash_Pos], 0
4376 0000B59A 76A0                <1>       jna    short pass_ppn_cdir
4377                              <1>
4378 0000B59C 8B0D[DA920100]      <1>       mov    ecx, [Last_Slash_Pos]
4379 0000B5A2 8B35[D6920100]      <1>       mov    esi, [First_Path_Pos]
4380 0000B5A8 29F1                <1>       sub    ecx, esi
4381 0000B5AA 41                  <1>       inc    ecx
4382                              <1>       ;cmp   ecx, 64
4383 0000B5AB 80F940              <1>       cmp    cl, 64
4384 0000B5AE 7715                <1>       ja     short loc_ppn_invalid_drive_stc
4385                              <1>
4386 0000B5B0 89F8                <1>       mov    eax, edi ; Dest Dir String Location (65 byte)
4387 0000B5B2 F3A4                <1>       rep    movsb
4388                              <1>       ;mov   [edi], cl ; 0, End of Dir String
4389 0000B5B4 8B35[DA920100]      <1>       mov    esi, [Last_Slash_Pos]
4390 0000B5BA 46                  <1>       inc    esi
4391 0000B5BB 89C7                <1>       mov    edi, eax
4392 0000B5BD AC                  <1>       lodsb
4393 0000B5BE 3C21                <1>       cmp    al, 21h
4394 0000B5C0 7381                <1>       jnb    short loc_ppn_get_filename
4395                              <1> loc_ppn_clc_return:
4396                              <1>       ;clc
4397 0000B5C2 31C0                <1>       xor    eax, eax
4398 0000B5C4 C3                  <1>       retn
4399                              <1>
4400                              <1> loc_ppn_invalid_drive_stc:
4401 0000B5C5 F5                  <1>       cmc    ; stc
4402                              <1> loc_ppn_invalid_drive:
4403                              <1>       ; cf = 1
4404                              <1>       ; The Drive Letter/Char < "A" or > "Z"
4405 0000B5C6 66B80F00            <1>       mov    ax, 0Fh
4406                              <1>       ; MS-DOS Error Code 0Fh = Disk Drive Invalid
4407                              <1>       ; (MainProg ErrMsg: "Drive not ready or read error!")
4408 0000B5CA C3                  <1>       retn
4409                              <1>
4410                              <1> find_longname:
4411                              <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
4412                              <1>       ; 24/01/2010 (DIR.ASM, 'proc_find_longname')
4413                              <1>       ; 17/10/2009
4414                              <1>
4415                              <1>       ; INPUT ->
4416                              <1>       ;      ESI = DOS short file name address
4417                              <1>       ;      for example: "filename.ext"
4418                              <1>       ;
4419                              <1>       ; OUTPUT ->
4420                              <1>       ;      ESI = ASCIIZ longname address (cf = 0)
4421                              <1>       ;      cf = 1 -> error number returns in EAX (AL)
4422                              <1>       ;      AL = 0 & CF=1 -> longname not found
4423                              <1>       ;           the file/directory has no longname
4424                              <1>       ;      cf = 0 -> AL = FAT Type
4425                              <1>
4426                              <1>       ; 17/10/2009
4427                              <1>       ; ASCIIZ string will be returned
4428                              <1>       ; as LongFileName
4429                              <1>       ; clearing/reset is not needed
4430                              <1>       ;mov   ecx, 33
4431                              <1>       ;mov   edi, LongFileName
4432                              <1>       ;sub   ax, ax ; 0
4433                              <1>       ;rep   stosw
4434                              <1>
4435                              <1>       ;mov   byte [LongNameFound], 0
4436                              <1>
4437                              <1>       ; ESI = ASCIIZ file/directory name address
4438                              <1>       ;   AL = Attributes AND mask
4439                              <1>       ;      (Result of AND must be equal to AL)
4440                              <1>       ;   AH = Negative attributes mask
4441                              <1>       ;      (Result of AND must be ZERO)
4442 0000B5CB 66B80008            <1>       mov    ax, 0800h
4443                              <1>            ; it must not be volume name or longname
4444 0000B5CF E87DDDFFFF          <1>       call   find_first_file
4445 0000B5D4 7216                <1>       jc     short loc_fln_retn
4446                              <1>
4447                              <1> loc_fln_check_FAT_Type:
4448 0000B5D6 803D[B5890100]01    <1>       cmp    byte [Current_FATType], 1
4449 0000B5DD 7306                <1>       jnb    short loc_fln_check_longname_yes_sign
4450                              <1>
4451 0000B5DF E839000000          <1>       call   get_fs_longname
4452 0000B5E4 C3                  <1>       retn
4453                              <1>
4454                              <1> loc_fln_check_longname_yes_sign:
4455 0000B5E5 08FF                <1>       or     bh, bh
4456 0000B5E7 7504                <1>       jnz    short loc_fln_check_longnamefound_number
4457                              <1> loc_fln_longname_not_found_retn:
4458 0000B5E9 31C0                <1>       xor    eax, eax
4459                              <1>       ; cf = 1 & al = 0 -> longname not found
4460 0000B5EB F9                  <1>       stc
4461                              <1> loc_fln_retn:
4462 0000B5EC C3                  <1>       retn
4463                              <1>
4464                              <1> loc_fln_check_longnamefound_number:
4465                              <1>       ; 'LongNameFound' is set by
4466                              <1>         ; by 'save_longname_sub_component'
4467                              <1>       ; which is called from
4468                              <1>       ; 'find_directory_entry'
4469                              <1>       ; which is called from
4470                              <1>       ; 'find_first_file'
4471                              <1>       ; It must 1 if the longname is valid
4472 0000B5ED 803D[B1910100]01    <1>         cmp     byte [LongNameFound], 1
4473 0000B5F4 75F3                <1>       jne    short loc_fln_longname_not_found_retn
4474                              <1>
4475                              <1> loc_fln_calculate_checksum:
4476 0000B5F6 E813000000          <1>       call   calculate_checksum
4477                              <1>       ; AL = shortname checksum
4478                              <1>
4479                              <1> loc_fln_longname_validation:
```

```
                                <1>       ; 'LFN_CheckSum' has been set already
                                <1>       ; by 'save_longname_sub_component'
                                <1>       ; which is called from
                                <1>       ; 'find_directory_entry'
                                <1>       ; which is called from
                                <1>       ; 'find_first_file'
0000B5FB 3805[B3910100]         <1>       cmp    [LFN_CheckSum], al
0000B601 75E6                   <1>       jne    short loc_fln_longname_not_found_retn
                                <1>
0000B603 BE[B4910100]           <1>       mov    esi, LongFileName
0000B608 A0[B5890100]           <1>       mov    al, [Current_FATType]
0000B60D C3                     <1>       retn
                                <1>
                                <1> calculate_checksum:
                                <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 17/10/2009 (DIR.ASM, 'proc_calculate_checksum')
                                <1>         ;
                                <1>       ; INPUT ->
                                <1>       ;     ESI = 11 byte DOS File Name location
                                <1>       ;     (in DOS Directory Entry Format)
                                <1>       ; OUTPUT ->
                                <1>       ;      AL = 8 bit checksum (CRC) value
                                <1>       ;
                                <1>       ; (Modified registers: EAX, ECX, ESI)
                                <1>
                                <1>       ; Erdogan Tan [ 17-10-2009 ]
                                <1>       ;  'ror al, 1' instruction
                                <1>
                                <1>       ; Erdogan Tan [ 20-06-2004 ]
                                <1>       ; This 8086 assembly code is an original code
                                <1>       ; which is adapted from C code in
                                <1>       ; Microsoft FAT32 File System Specification
                                <1>       ; Version 1.03, December 6, 2000
                                <1>       ; Page 28
                                <1>
0000B60E 30C0                   <1>       xor    al, al
0000B610 B90B000000             <1>       mov    ecx, 11
                                <1> loc_next_sum:
                                <1>       ;xor   ah, ah
                                <1>       ;test  al, 1
                                <1>       ;jz    short pass_ah_80h
                                <1>       ;mov   ah, 80h
                                <1> ;pass_ah_80h:
                                <1>       ;shr   al, 1
0000B615 D0C8                   <1>       ror    al, 1 ; 17/10/2009
0000B617 0206                   <1>       add    al, [esi]
0000B619 46                     <1>       inc    esi
                                <1>       ;add   al, ah
0000B61A E2F9                   <1>       loop   loc_next_sum
0000B61C C3                     <1>       retn
                                <1>
                                <1> get_fs_longname:
                                <1>       ; temporary (13/02/2016)
0000B61D 31C0                   <1>       xor eax, eax
0000B61F F9                     <1>       stc
0000B620 C3                     <1>       retn
                                <1>
                                <1> make_sub_directory:
                                <1>       ; 16/10/2016
                                <1>       ; 02/03/2016, 03/03/2016
                                <1>       ; 26/02/2016, 27/02/2016
                                <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 01/08/2011 (DIR.ASM, 'proc_make_directory')
                                <1>       ; 10/07/2010
                                <1>       ; INPUT ->
                                <1>       ;     ESI = ASCIIZ Directory Name
                                <1>       ;     CL = Directory Attributes
                                <1>       ; OUTPUT ->
                                <1>       ;     EAX = New sub dir's first cluster
                                <1>       ;     ESI = Logical Dos Drv Descr. Table Addr.
                                <1>       ;     CF = 1 -> error code in AL (EAX)
                                <1>
                                <1>       ;test  cl, 10h  ; directory
                                <1>       ;jz    short loc_make_directory_access_denied
                                <1>       ;test  cl, 08h ; volume name
                                <1>       ;jnz   short loc_make_directory_access_denied
                                <1>
0000B621 80E107                 <1>       and    cl, 07h
0000B624 880D[30930100]         <1>       mov    byte [mkdir_attrib], cl
                                <1>
0000B62A 56                     <1>       push   esi
0000B62B 31DB                   <1>       xor    ebx, ebx
0000B62D 8A3D[B6890100]         <1>       mov    bh, [Current_Drv]
0000B633 BE00010900             <1>       mov    esi, Logical_DOSDisks
0000B638 01DE                   <1>       add    esi, ebx
0000B63A 5B                     <1>       pop    ebx
                                <1>
                                <1>       ; 10/07/2010 -> 1st writable disk check for trdos
                                <1>       ; LD_DiskType = 0 for write protection (read only)
0000B63B 807E0101               <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
0000B63F 730B                   <1>       jnb    short loc_mkdir_check_file_sytem
                                <1>       ; 16/10/2016 (13h -> 30)
0000B641 B81E000000             <1>       mov    eax, 30 ; 'Disk write-protected' error
0000B646 BA00000000             <1>       mov    edx, 0
                                <1>       ; err retn: EDX = 0, EBX = Dir name offset
                                <1>       ;ESI = Logical DOS drive description table address
0000B64B C3                     <1>       retn
                                <1>
                                <1> ;loc_make_directory_access_denied:
                                <1>       ;mov   ax, 05h ; access denied (invalid attributes input)
                                <1>       ;stc
                                <1>       ;retn
                                <1>
                                <1> loc_mkdir_check_file_sytem:
0000B64C 807E0301               <1>       cmp    byte [esi+LD_FATType], 1
```

```
4585 0000B650 730B              <1>       jnb    short loc_mkdir_check_free_sectors
4586                            <1>
4587                            <1> loc_make_fs_directory:
4588 0000B652 A1[B0890100]      <1>       mov    eax, [Current_Dir_FCluster]
4589                            <1>       ; EAX = Parent directory DDT Address
4590                            <1>       ; ESI = Logical DOS Drive DT Address
4591                            <1>       ; EBX = Directory name offset (as ASCIIZ name)
4592 0000B657 E8D5150000        <1>       call   make_fs_directory
4593 0000B65C C3                <1>       retn
4594                            <1>
4595                            <1> loc_mkdir_check_free_sectors:
4596 0000B65D 0FB64613          <1>         movzx   eax, byte [esi+LD_BPB+SecPerClust]
4597 0000B661 8B4E74            <1>       mov    ecx, [esi+LD_FreeSectors]
4598 0000B664 39C1              <1>       cmp    ecx, eax
4599 0000B666 7255              <1>       jb     short loc_mkdir_insufficient_disk_space
4600                            <1>
4601                            <1> loc_make_fat_directory:
4602 0000B668 891D[20930100]    <1>       mov    [mkdir_DirName_Offset], ebx
4603 0000B66E 890D[2C930100]    <1>       mov    [mkdir_FreeSectors], ecx
4604                            <1>
4605                            <1>       ;mov   al, [esi+LD_BPB+SecPerClust]
4606 0000B674 A2[32930100]      <1>       mov    byte [mkdir_SecPerClust], al
4607                            <1>
4608                            <1> loc_mkdir_gffc_1:
4609 0000B679 E80F180000        <1>       call   get_first_free_cluster
4610 0000B67E 722A              <1>       jc     short loc_mkdir_gffc_retn
4611                            <1>
4612                            <1> ;loc_mkdir_gffc_1_cont:
4613                            <1>       ;cmp   eax, 2
4614                            <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
4615                            <1>
4616                            <1> ;loc_mkdir_gffc_1_save_fcluster:
4617 0000B680 A3[24930100]      <1>       mov    [mkdir_FFCluster], eax
4618                            <1>
4619                            <1> loc_mkdir_locate_ffe:
4620                            <1>       ; Current directory fcluster <> Directory buffer cluster
4621                            <1>       ; Current directory will be reloaded by
4622                            <1>       ; 'locate_current_dir_file' procedure
4623                            <1>       ;
4624                            <1>       ; ESI = Logical DOS Drive Description Table Address
4625                            <1>       ;push  esi ; 27/02/2016
4626 0000B685 31C0              <1>       xor    eax, eax
4627 0000B687 89C1              <1>         mov  ecx, eax
4628 0000B689 6649              <1>       dec    cx ; FFFFh
4629                            <1>       ; CX = FFFFh -> find first deleted or free entry
4630                            <1>       ; ESI would be ASCIIZ filename address if the call
4631                            <1>       ; would not be for first free or deleted dir entry
4632 0000B68B E8CFFAFFFF        <1>       call   locate_current_dir_file
4633 0000B690 734C              <1>       jnc    short loc_mkdir_set_ff_dir_entry_1
4634                            <1>       ;pop   esi
4635                            <1>       ; ESI = Logical DOS Drive Description Table Address
4636 0000B692 83F802            <1>       cmp    eax, 2  ; cmp al, 2 ; File/Dir not found !
4637 0000B695 752B              <1>       jne    short loc_mkdir_stc_return
4638                            <1>
4639                            <1> loc_mkdir_add_new_cluster:
4640 0000B697 3805[B5890100]    <1>       cmp    byte [Current_FATType], al ; 2
4641                            <1>       ;cmp   byte ptr [esi+LD_FATType], 2
4642 0000B69D 770C              <1>       ja     short loc_mkdir_add_new_cluster_check_fsc
4643 0000B69F 803D[B4890100]01  <1>       cmp    byte [Current_Dir_Level], 1
4644                            <1>       ;cmp   byte [esi+LD_CDirLevel], 1
4645 0000B6A6 7303              <1>       jnb    short loc_mkdir_add_new_cluster_check_fsc
4646                            <1>
4647 0000B6A8 B00C              <1>       mov    al, 12 ; No more files
4648                            <1> loc_mkdir_gffc_retn:
4649 0000B6AA C3                <1>       retn
4650                            <1>
4651                            <1> loc_mkdir_add_new_cluster_check_fsc:
4652 0000B6AB 8B0D[2C930100]    <1>       mov    ecx, [mkdir_FreeSectors]
4653                            <1>       ;movzx eax, byte [mkdir_SecPerClust]
4654 0000B6B1 A0[32930100]      <1>       mov    al, [mkdir_SecPerClust]
4655 0000B6B6 66D1E0            <1>       shl    ax, 1 ; AX = 2 * AX
4656 0000B6B9 39C1              <1>       cmp    ecx, eax
4657 0000B6BB 7350              <1>       jnb    short loc_mkdir_add_new_subdir_cluster
4658                            <1>
4659                            <1> loc_mkdir_insufficient_disk_space:
4660                            <1>       ;mov   edx, ecx
4661                            <1> ;loc_mkdir_gffc_insufficient_disk_space:
4662 0000B6BD 66B82700          <1>       mov    ax, 27h ; MSDOS err => insufficient disk space
4663                            <1>       ; err retn: EDX = Free sectors, EBX = Dir name offset
4664                            <1>         ; ESI -> Dos drive description table address
4665                            <1>       ;; ecx = edx
4666                            <1>       ;
4667 0000B6C1 C3                <1>       retn
4668                            <1>
4669                            <1> loc_mkdir_stc_return:
4670 0000B6C2 F9                <1>       stc
4671 0000B6C3 C3                <1>       retn
4672                            <1>
4673                            <1> loc_mkdir_gffc_2:
4674 0000B6C4 E8C4170000        <1>       call   get_first_free_cluster
4675 0000B6C9 72DF              <1>       jc     short loc_mkdir_gffc_retn
4676                            <1>
4677                            <1> ;loc_mkdir_gffc_1_cont:
4678                            <1>       ;cmp   eax, 2
4679                            <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
4680                            <1>
4681                            <1> ;loc_mkdir_gffc_2_save_fcluster:
4682 0000B6CB A3[24930100]      <1>       mov    [mkdir_FFCluster], eax
4683                            <1>
4684 0000B6D0 A1[28930100]      <1>       mov    eax, [mkdir_LastDirCluster]
4685                            <1>
4686 0000B6D5 E842170000        <1>       call   load_FAT_sub_directory
4687 0000B6DA 72CE              <1>       jc     short loc_mkdir_gffc_retn
4688                            <1>
4689 0000B6DC 31FF              <1>       xor    edi, edi
```

```
                            <1> loc_mkdir_set_ff_dir_entry_1:
                            <1>       ; 27/02/2016
0000B6DE 56                 <1>       push   esi ; Logical DOS Drv Desc. Tbl. address
                            <1>       ; EDI = Directory Entry Address
0000B6DF 8B35[20930100]     <1>       mov    esi, [mkdir_DirName_Offset]
0000B6E5 A1[24930100]       <1>       mov    eax, [mkdir_FFCluster]
                            <1>
0000B6EA 66B91000           <1>       mov    cx, 10h      ; CL = Directory attribute
                            <1>                   ; CH = 0 -> File size is 0
0000B6EE 0A0D[30930100]     <1>       or     cl, [mkdir_attrib] ; S, H, R
0000B6F4 E8B0010000         <1>       call   make_directory_entry
                            <1>
0000B6F9 5E                 <1>       pop    esi
                            <1>
0000B6FA C605[DC900100]02   <1>       mov    byte [DirBuff_ValidData], 2
0000B701 E880020000         <1>       call   save_directory_buffer
0000B706 0F83DA000000       <1>         jnc     loc_mkdir_set_ff_dir_entry_2
                            <1>
                            <1> loc_mkdir_return:
0000B70C C3                 <1>       retn
                            <1>
                            <1> loc_mkdir_add_new_subdir_cluster:
0000B70D 8B15[E1900100]     <1>       mov    edx, [DirBuff_Cluster]
0000B713 8915[28930100]     <1>       mov    [mkdir_LastDirCluster], edx
                            <1>
0000B719 A1[24930100]       <1>       mov    eax, [mkdir_FFCluster]
0000B71E E8F9160000         <1>       call   load_FAT_sub_directory
0000B723 72E7               <1>       jc     short loc_mkdir_return
                            <1>       ; eax = 0
                            <1>       ; ecx =  directory buffer sector count (<= 128)
                            <1>
                            <1> pass_mkdir_add_new_subdir_cluster:
0000B725 29FF               <1>       sub    edi, edi ; 0
                            <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
0000B727 668B4611           <1>       mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
0000B72B 66C1E802           <1>       shr    ax, 2 ; 'byte count / 4' for 'stosd'
0000B72F 66F7E1             <1>       mul    cx ; max = 128*(512/4) -> 16384 (stosd)
0000B732 6689C1             <1>       mov    cx, ax
0000B735 6629C0             <1>       sub    ax, ax ; 0
0000B738 F3AB               <1>       rep    stosd ; clear directory buffer
                            <1>
0000B73A C605[DC900100]02   <1>       mov    byte [DirBuff_ValidData], 2
0000B741 E840020000         <1>       call   save_directory_buffer
0000B746 72C4               <1>       jc     short loc_mkdir_return
                            <1>
                            <1> loc_mkdir_save_added_cluster:
0000B748 A1[28930100]       <1>       mov    eax, [mkdir_LastDirCluster]
0000B74D 8B0D[24930100]     <1>       mov    ecx, [mkdir_FFCluster]
                            <1>       ; 01/03/2016
0000B753 31D2               <1>       xor    edx, edx
0000B755 8915[D2900100]     <1>       mov    [FAT_ClusterCounter], edx ; 0 ; reset
0000B75B E800180000         <1>       call   update_cluster
0000B760 7304               <1>       jnc    short loc_mkdir_save_fat_buffer_0
0000B762 09C0               <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
0000B764 7518               <1>       jnz    short loc_mkdir_save_fat_buffer_stc_retn
                            <1>
                            <1> loc_mkdir_save_fat_buffer_0:
0000B766 A1[24930100]       <1>       mov    eax, [mkdir_FFCluster]
0000B76B A3[28930100]       <1>       mov    [mkdir_LastDirCluster], eax
                            <1>
0000B770 31C9               <1>       xor    ecx, ecx
0000B772 49                 <1>       dec    ecx ; FFFFFFFFh
                            <1>       ; ESI = Logical DOS Drive Description Table address
0000B773 E8E8170000         <1>       call   update_cluster
0000B778 731A               <1>       jnc    short loc_mkdir_save_fat_buffer_1
0000B77A 09C0               <1>       or     eax, eax
0000B77C 7416               <1>       jz     short loc_mkdir_save_fat_buffer_1
                            <1>
                            <1> loc_mkdir_save_fat_buffer_stc_retn:
                            <1>       ; 01/03/2016
0000B77E 803D[D2900100]01   <1>       cmp    byte [FAT_ClusterCounter], 1
0000B785 720C               <1>       jb     short loc_mkdir_save_fat_buffer_retn
                            <1>
0000B787 66BB00FF           <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
                            <1>                         ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
0000B78B 50                 <1>       push   eax
0000B78C E8211B0000         <1>       call   calculate_fat_freespace
0000B791 58                 <1>       pop    eax
0000B792 F9                 <1>       stc
                            <1> loc_mkdir_save_fat_buffer_retn:
0000B793 C3                 <1>       retn
                            <1>
                            <1> loc_mkdir_save_fat_buffer_1:
                            <1>       ; byte [FAT_BuffValidData] = 2
0000B794 E8841A0000         <1>       call   save_fat_buffer
0000B799 72E3               <1>       jc     short loc_mkdir_save_fat_buffer_stc_retn
                            <1>
                            <1>       ; 01/03/2016
0000B79B 803D[D2900100]01   <1>       cmp    byte [FAT_ClusterCounter], 1
0000B7A2 721B               <1>       jb     short loc_mkdir_save_fat_buffer_2
                            <1>
                            <1>       ; ESI = Logical DOS Drive Description Table address
0000B7A4 A1[D2900100]       <1>       mov    eax, [FAT_ClusterCounter]
0000B7A9 66BB01FF           <1>       mov    bx, 0FF01h ; add free clusters
0000B7AD E8001B0000         <1>       call   calculate_fat_freespace
                            <1>
                            <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
                            <1>       ;jnz   short loc_mkdir_save_fat_buffer_2
                            <1>
                            <1>       ; ecx > 0 -> Recalculation is needed
0000B7B2 09C9               <1>       or     ecx, ecx
0000B7B4 7409               <1>       jz     short loc_mkdir_save_fat_buffer_2
```

```
                                <1>
0000B7B6 66BB00FF               <1>      mov    bx, 0FF00h ; ; recalculate free space
0000B7BA E8F31A0000             <1>      call   calculate_fat_freespace
                                <1>
                                <1> loc_mkdir_save_fat_buffer_2:
0000B7BF C605[33930100]01       <1>      mov    byte [mkdir_add_new_cluster], 1
0000B7C6 E9C4000000             <1>      jmp    loc_mkdir_upd_parent_dir_lmdt
                                <1>
                                <1> loc_mkdir_update_sub_dir_cluster:
0000B7CB A1[24930100]           <1>      mov    eax, [mkdir_FFCluster]
0000B7D0 29C9                   <1>      sub    ecx, ecx ; 0
                                <1>      ; 01/03/2016
0000B7D2 890D[D2900100]         <1>      mov    [FAT_ClusterCounter], ecx ; 0 ; Reset
0000B7D8 49                     <1>      dec    ecx ; 0FFFFFFFFh
                                <1>
                                <1>      ; ESI = Logical DOS Drive Desciption Table address
0000B7D9 E882170000             <1>      call   update_cluster
0000B7DE 7379                   <1>      jnc    short loc_mkdir_save_fat_buffer_3
0000B7E0 09C0                   <1>      or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
0000B7E2 7475                   <1>      jz     short loc_mkdir_save_fat_buffer_3
                                <1>      ; 01/03/2016
0000B7E4 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
0000B7E6 A1[24930100]           <1>      mov    eax, [mkdir_FFCluster]
                                <1>      ; Load disk sectors as a directory cluster
0000B7EB E82C160000             <1>      call   load_FAT_sub_directory
0000B7F0 7266                   <1>      jc     short retn_make_fat_directory
                                <1>
                                <1>      ; eax = 0
                                <1>      ; ecx =  directory buffer sector count (<= 128)
                                <1>
0000B7F2 BF40000800             <1>      mov    edi, Directory_Buffer + 64 ; 26/02/2016
                                <1>
                                <1>      ; 02/03/2016
0000B7F7 668B4611               <1>      mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
0000B7FB 66C1E802               <1>      shr    ax, 2 ; 'byte count / 4' for 'stosd'
0000B7FF F7E1                   <1>      mul    ecx
0000B801 89C1                   <1>      mov    ecx, eax
0000B803 6629C0                 <1>      sub    ax, ax
0000B806 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>
0000B808 BF00000800             <1>      mov    edi, Directory_Buffer ; 26/02/2016
                                <1>
0000B80D 56                     <1>      push   esi
                                <1>
0000B80E BE[34930100]           <1>      mov    esi, mkdir_Name
0000B813 66C7062E00             <1>      mov    word [esi], 2Eh ; db '.', '0'
                                <1>
0000B818 A1[24930100]           <1>      mov    eax, [mkdir_FFCluster]
0000B81D 66B91000               <1>      mov    cx, 10h ; CL = Directory attribute
                                <1>                     ; CH = 0 -> File size is 0
0000B821 E883000000             <1>      call   make_directory_entry
                                <1>
0000B826 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>
0000B82B 29C0                   <1>      sub    eax, eax
0000B82D 3805[B4890100]         <1>      cmp    byte [Current_Dir_Level], al ; 0
0000B833 7605                   <1>      jna    short loc_mkdir_set_ff_dir_entry_3
0000B835 A1[B0890100]           <1>      mov    eax, [Current_Dir_FCluster] ; parent dir
                                <1> loc_mkdir_set_ff_dir_entry_3:
0000B83A 66C746012E00           <1>      mov    word [esi+1], 2Eh ; db '.', '0'
                                <1>
                                <1>      ;mov   cx, 10h
0000B840 E864000000             <1>      call   make_directory_entry
                                <1>
0000B845 5E                     <1>      pop    esi
                                <1>
0000B846 C605[DC900100]02       <1>      mov    byte [DirBuff_ValidData], 2
0000B84D E834010000             <1>      call   save_directory_buffer
0000B852 0F8373FFFFFF           <1>        jnc     loc_mkdir_update_sub_dir_cluster
                                <1>
                                <1> retn_make_fat_directory:
0000B858 C3                     <1>      retn
                                <1>
                                <1> loc_mkdir_save_fat_buffer_3:
                                <1>      ; 01/03/2016
                                <1>      ; byte [FAT_BuffValidData] = 2
0000B859 E8BF190000             <1>      call   save_fat_buffer
0000B85E 0F821AFFFFFF           <1>        jc      loc_mkdir_save_fat_buffer_stc_retn
                                <1>
0000B864 803D[D2900100]01       <1>      cmp    byte [FAT_ClusterCounter], 1
0000B86B 721B                   <1>      jb     short loc_mkdir_save_fat_buffer_4
```

```
4900                                    <1>
4901                                    <1>     ; ESI = Logical DOS Drive Description Table address
4902 0000B86D A1[D2900100]              <1>     mov   eax, [FAT_ClusterCounter]
4903 0000B872 66BB01FF                  <1>     mov   bx, 0FF01h ; add free clusters
4904 0000B876 E8371A0000                <1>     call  calculate_fat_freespace
4905                                    <1>
4906                                    <1>     ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
4907                                    <1>       ;jnz    short loc_mkdir_save_fat_buffer_4
4908                                    <1>
4909                                    <1>     ; ecx > 0 -> Recalculation is needed
4910 0000B87B 09C9                     <1>     or     ecx, ecx
4911 0000B87D 7409                     <1>       jz      short loc_mkdir_save_fat_buffer_4
4912                                    <1>
4913 0000B87F 66BB00FF                 <1>     mov   bx, 0FF00h ; recalculate free space
4914 0000B883 E82A1A0000               <1>     call  calculate_fat_freespace
4915                                    <1>
4916                                    <1> loc_mkdir_save_fat_buffer_4:
4917 0000B888 C605[33930100]00         <1>     mov   byte [mkdir_add_new_cluster], 0
4918                                    <1>
4919                                    <1> loc_mkdir_upd_parent_dir_lmdt:
4920 0000B88F E88D010000               <1>     call  update_parent_dir_lmdt
4921                                    <1>
4922                                    <1>     ; 01/03/2016
4923 0000B894 803D[33930100]00         <1>     cmp   byte [mkdir_add_new_cluster], 0
4924 0000B89B 0F8723FEFFFF             <1>       ja      loc_mkdir_gffc_2
4925                                    <1>
4926                                    <1> loc_mkdir_retn_new_dir_cluster:
4927 0000B8A1 A1[24930100]             <1>     mov   eax, [mkdir_FFCluster]
4928 0000B8A6 31D2                     <1>     xor   edx, edx
4929                                    <1> loc_mkdir_retn:
4930 0000B8A8 C3                       <1>     retn
4931                                    <1>
4932                                    <1> make_directory_entry:
4933                                    <1>     ; 02/03/2016
4934                                    <1>     ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
4935                                    <1>     ; 09/08/2010 (DIR.ASM, 'proc_make_directory_entry')
4936                                    <1>     ; 17/07/2010
4937                                    <1>     ; INPUT ->
4938                                    <1>     ;     EDI = Directory Entry Address
4939                                    <1>     ;     ESI = Dot File Name Location
4940                                    <1>     ;     EAX = First Cluster
4941                                    <1>     ;     File Size = 0 (Must be set later)
4942                                    <1>     ;     CL = Attributes
4943                                    <1>     ;     CH = 0 (File size = 0)
4944                                    <1>     ;     (If CH>0, File size is in dword [EBX]) (*)
4945                                    <1>     ; OUTPUT ->
4946                                    <1>     ;     EDI = Directory Entry Address
4947                                    <1>     ;     ESI = Dot File Name Location (Capitalized)
4948                                    <1>     ;     If CH input = 0, File Size = 0
4949                                    <1>     ;     Otherwise file size is as dword [EBX] (*)
4950                                    <1>     ;     DX = Date, AX = Time in DOS Dir Entry format
4951                                    <1>     ;     EBX = same
4952                                    <1>     ;     ECX = same
4953                                    <1>
4954 0000B8A9 51                       <1>     push  ecx
4955                                    <1>
4956 0000B8AA 884F0B                   <1>     mov   [edi+11], cl ; Attributes
4957 0000B8AD 6689471A                 <1>     mov   [edi+26], ax ; FClusterLw, 26
4958 0000B8B1 C1E810                   <1>     shr   eax, 16
4959 0000B8B4 66894714                 <1>     mov   [edi+20], ax ; FClusterHw, 20
4960 0000B8B8 6631C0                   <1>     xor   ax, ax
4961 0000B8BB 6689470C                 <1>     mov   [edi+12], ax ; NTReserved, 12
4962                                    <1>                        ; CrtTimeTenth, 13
4963 0000B8BF 08ED                     <1>     or    ch, ch
4964 0000B8C1 7402                     <1>     jz    short loc_make_direntry_set_filesize
4965                                    <1>
4966 0000B8C3 8B03                     <1>     mov   eax, [ebx]
4967                                    <1>
4968                                    <1> loc_make_direntry_set_filesize:
4969 0000B8C5 89471C                   <1>     mov   [edi+28], eax ; FileSize, 28
4970                                    <1>
4971 0000B8C8 E88AFBFFFF               <1>     call  convert_file_name
4972                                    <1>     ;EDI = Dir Entry Format File Name Location
4973                                    <1>     ;ESI = Dot File Name Location (capitalized)
4974                                    <1>
4975 0000B8CD E816000000               <1>     call  convert_current_date_time
4976                                    <1>     ; OUTPUT -> DX = Date in dos dir entry format
4977                                    <1>       ;       AX = Time in dos dir entry format
4978 0000B8D2 6689470E                 <1>     mov   [edi+14], ax ; CrtTime, 14
4979 0000B8D6 66895710                 <1>     mov   [edi+16], dx ; CrtDate, 16
4980 0000B8DA 66895712                 <1>     mov   [edi+18], dx ; LastAccDate, 18
4981 0000B8DE 66894716                 <1>     mov   [edi+22], ax ; WrtTime, 14
4982 0000B8E2 66895718                 <1>     mov   [edi+24], dx ; WrtDate, 16
4983 0000B8E6 59                       <1>     pop   ecx
4984                                    <1>
4985 0000B8E7 C3                       <1>     retn
4986                                    <1>
4987                                    <1> convert_current_date_time:
4988                                    <1>     ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
4989                                    <1>     ; 13/06/2010 (DIR.ASM, 'proc_convert_current_date_time')
4990                                    <1>     ; converts date&time to dos dir entry format
4991                                    <1>     ; INPUT -> none
4992                                    <1>     ; OUTPUT -> DX = Date in dos dir entry format
4993                                    <1>     ;           AX = Time in dos dir entry format
4994                                    <1>
4995 0000B8E8 B404                     <1>     mov   ah, 04h ; Return Current Date
4996 0000B8EA E81BB0FFFF               <1>     call  int1Ah
4997                                    <1>
4998 0000B8EF 88E8                     <1>     mov   al, ch ; <- century BCD
4999 0000B8F1 240F                     <1>     and   al, 0Fh
5000 0000B8F3 88EC                     <1>     mov   ah, ch
5001 0000B8F5 C0EC04                   <1>     shr   ah, 4
5002 0000B8F8 D50A                     <1>     aad
5003 0000B8FA 88C5                     <1>     mov   ch, al ; -> century
5004                                    <1>
```

```
5005 0000B8FC 88C8              <1>       mov   al, cl ; <- year BCD
5006 0000B8FE 240F              <1>       and   al, 0Fh
5007 0000B900 88CC              <1>       mov   ah, cl
5008 0000B902 C0EC04            <1>       shr   ah, 4
5009 0000B905 D50A              <1>       aad
5010 0000B907 88C1              <1>       mov   cl, al ; -> year
5011                            <1>
5012 0000B909 88E8              <1>       mov   al, ch
5013 0000B90B B464              <1>       mov   ah, 100
5014 0000B90D F6E4              <1>       mul   ah
5015 0000B90F 30ED              <1>       xor   ch, ch
5016 0000B911 6601C8            <1>       add   ax, cx
5017 0000B914 662DBC07          <1>       sub   ax, 1980 ; ms-dos epoch
5018 0000B918 6689C1            <1>       mov   cx, ax
5019                            <1>
5020 0000B91B 88F0              <1>       mov   al, dh ; <- month in bcd
5021 0000B91D 240F              <1>       and   al, 0Fh
5022 0000B91F 88F4              <1>       mov   ah, dh
5023 0000B921 C0EC04            <1>       shr   ah, 4
5024 0000B924 D50A              <1>       aad
5025 0000B926 88C6              <1>       mov   dh, al ; -> month
5026                            <1>
5027 0000B928 88D0              <1>       mov   al, dl ; <- day BCD
5028 0000B92A 240F              <1>       and   al, 0Fh
5029 0000B92C 88D4              <1>       mov   ah, dl
5030 0000B92E C0EC04            <1>       shr   ah, 4
5031 0000B931 D50A              <1>       aad
5032 0000B933 88C2              <1>       mov   dl, al ; -> day
5033                            <1>
5034 0000B935 88C8              <1>       mov   al, cl ; count of years from 1980
5035 0000B937 66C1E004          <1>       shl   ax, 4
5036 0000B93B 08F0              <1>       or    al, dh ; month of year, 1 to 12
5037 0000B93D 66C1E005          <1>       shl   ax, 5
5038 0000B941 08D0              <1>       or    al, dl ; day of year, 1 to 31
5039                            <1>
5040 0000B943 6650              <1>       push  ax ; push date
5041                            <1>
5042 0000B945 B402              <1>       mov   ah, 02h ; Return Current Time
5043 0000B947 E8BEAFFFFF        <1>       call  int1Ah
5044                            <1>
5045 0000B94C 88E8              <1>       mov   al, ch ; <- hours BCD
5046 0000B94E 240F              <1>       and   al, 0Fh
5047 0000B950 88EC              <1>       mov   ah, ch
5048 0000B952 C0EC04            <1>       shr   ah, 4
5049 0000B955 D50A              <1>       aad
5050 0000B957 88C5              <1>       mov   ch, al ; -> hours
5051                            <1>
5052 0000B959 88C8              <1>       mov   al, cl ; <- minutes BCD
5053 0000B95B 240F              <1>       and   al, 0Fh
5054 0000B95D 88CC              <1>       mov   ah, cl
5055 0000B95F C0EC04            <1>       shr   ah, 4
5056 0000B962 D50A              <1>       aad
5057 0000B964 88C1              <1>       mov   cl, al ; -> minutes
5058                            <1>
5059 0000B966 88F0              <1>       mov   al, dh ; <- seconds BCD
5060 0000B968 240F              <1>       and   al, 0Fh
5061 0000B96A 88F4              <1>       mov   ah, dh
5062 0000B96C C0EC04            <1>       shr   ah, 4
5063 0000B96F D50A              <1>       aad
5064 0000B971 88C6              <1>       mov   dh, al ; -> seconds
5065                            <1>
5066 0000B973 88E8              <1>       mov   al, ch ; hours
5067 0000B975 66C1E006          <1>       shl   ax, 6
5068 0000B979 08C8              <1>       or    al, cl ; minutes
5069 0000B97B 66C1E005          <1>       shl   ax, 5
5070 0000B97F D0EE              <1>       shr   dh, 1 ; 2 seconds
5071                            <1>       ; There is a bug in TRDOS v1 here !
5072                            <1>       ; it was 'or al, dl' !
5073 0000B981 08F0              <1>       or    al, dh ; seconds
5074                            <1>
5075 0000B983 665A              <1>       pop   dx ; pop date
5076                            <1>
5077 0000B985 C3                <1>       retn
5078                            <1>
5079                            <1> save_directory_buffer:
5080                            <1>       ; 15/10/2016
5081                            <1>       ; 23/03/2016
5082                            <1>       ; 26/02/2016
5083                            <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
5084                            <1>       ; 01/08/2011
5085                            <1>       ; 14/03/2010
5086                            <1>       ; INPUT ->
5087                            <1>       ;       none
5088                            <1>       ; OUTPUT ->
5089                            <1>       ;  cf = 0 -> write OK...
5090                            <1>       ;  cf = 1 -> error code in AL (EAX)
5091                            <1>       ;  cf = 1 & AL = 0Dh => CH & CL = FS & FAT type
5092                            <1>       ;  EBX = Directory Buffer Address
5093                            <1>       ;
5094                            <1>       ;  (EAX, ECX, EDX will be modified)
5095                            <1>
5096 0000B986 BB00000800        <1>       mov   ebx, Directory_Buffer
5097 0000B98B 803D[DC900100]02  <1>       cmp   byte [DirBuff_ValidData], 2
5098 0000B992 7403              <1>       je    short loc_save_dir_buffer
5099 0000B994 31C0              <1>       xor   eax, eax
5100 0000B996 C3                <1>       retn
5101                            <1>
5102                            <1> loc_save_dir_buffer:
5103 0000B997 56                <1>       push  esi
5104 0000B998 31DB              <1>       xor   ebx, ebx
5105 0000B99A 8A3D[DA900100]    <1>         mov     bh, [DirBuff_DRV]
5106 0000B9A0 80EF41            <1>       sub   bh, 'A'
5107 0000B9A3 BE00010900        <1>         mov     esi, Logical_DOSDisks
5108 0000B9A8 01DE              <1>       add   esi, ebx
5109 0000B9AA 668B4E03          <1>         mov     cx, [esi+LD_FATType]
```

```
                                <1>       ; CH = FS Type (A1h for FS)
                                <1>       ; CL = FAT Type (0 for FS)
0000B9AE 08C9                   <1>       or     cl, cl
0000B9B0 7433                   <1>       jz     short loc_save_dir_buff_stc_retn
                                <1>
                                <1> loc_save_dir_buffer_check_cluster_no:
0000B9B2 A1[E1900100]           <1>       mov    eax, [DirBuff_Cluster]
0000B9B7 28FF                   <1>       sub    bh, bh ; ebx = 0
0000B9B9 09C0                   <1>       or     eax, eax
0000B9BB 7540                   <1>       jnz    short loc_save_sub_dir_buffer
0000B9BD 8A25[DB900100]         <1>       mov    ah, [DirBuff_FATType]
0000B9C3 FEC3                   <1>       inc    bl ;  bl = 1
0000B9C5 38DC                   <1>       cmp    ah, bl
0000B9C7 721D                   <1>       jb     short loc_save_dir_buff_inv_data_retn
0000B9C9 FEC3                   <1>       inc    bl ; bl = 2
0000B9CB 38E3                   <1>       cmp    bl, ah
0000B9CD 7217                   <1>       jb     short loc_save_dir_buff_inv_data_retn
                                <1>
                                <1> loc_save_root_dir_buffer:
0000B9CF 668B5E17               <1>       mov    bx, [esi+LD_BPB+RootDirEnts]
0000B9D3 6683C30F               <1>       add    bx, 15
0000B9D7 66C1EB04               <1>       shr    bx, 4 ; 16 dir entries per sector
0000B9DB 6609DB                 <1>       or     bx, bx
0000B9DE 7405                   <1>       jz     short loc_save_dir_buff_stc_retn
                                <1>       ;mov   ecx, ebx
0000B9E0 8B4664                 <1>       mov    eax, [esi+LD_ROOTBegin] ; 26/02/2016
0000B9E3 EB23                   <1>       jmp    short loc_write_directory_to_disk
                                <1>
                                <1> loc_save_dir_buff_stc_retn:
0000B9E5 F9                     <1>       stc
                                <1> loc_save_dir_buff_inv_data_retn:
                                <1>       ; 15/10/2016 (0Dh -> 29)
0000B9E6 B01D                   <1>       mov    al, 29 ; Invalid data !
0000B9E8 C605[DC900100]00       <1>       mov    byte [DirBuff_ValidData], 0
0000B9EF EB05                   <1>       jmp    short loc_save_dir_buff_retn
                                <1>
                                <1> loc_write_directory_to_disk_err:
                                <1>       ; 15/10/2016 (disk write error code, 1Dh -> 18)
0000B9F1 B812000000             <1>       mov    eax, 18 ; Drive not ready or write error
                                <1>
                                <1> loc_save_dir_buff_retn:
0000B9F6 BB00000800             <1>       mov    ebx, Directory_Buffer
0000B9FB 5E                     <1>       pop    esi
0000B9FC C3                     <1>       retn
                                <1>
                                <1> loc_save_sub_dir_buffer:
                                <1>       ; ebx  = 0
0000B9FD 83E802                 <1>       sub    eax, 2
0000BA00 8A5E13                 <1>       mov    bl, [esi+LD_BPB+SecPerClust]
0000BA03 F7E3                   <1>       mul    ebx
0000BA05 034668                 <1>         add     eax, [esi+LD_DATABegin]
                                <1>       ;mov   ecx, ebx
                                <1>
                                <1> loc_write_directory_to_disk:
0000BA08 89D9                   <1>       mov    ecx, ebx
0000BA0A BB00000800             <1>       mov    ebx, Directory_Buffer
0000BA0F E85A700000             <1>       call   disk_write
0000BA14 72DB                   <1>       jc     short loc_write_directory_to_disk_err
                                <1>
                                <1> loc_save_dir_buff_validate_retn:
0000BA16 C605[DC900100]01       <1>       mov    byte [DirBuff_ValidData], 1
0000BA1D 31C0                   <1>       xor    eax, eax
                                <1>       ; 26/02/2016
0000BA1F EBD5                   <1>       jmp    short loc_save_dir_buff_retn
                                <1>
                                <1> update_parent_dir_lmdt:
                                <1>       ; 29/12/2017
                                <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 01/08/2011
                                <1>       ; 16/10/2010
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      none
                                <1>       ; OUTPUT ->
                                <1>       ;      (last modification date & time of the parent dir
                                <1>       ;      will be changed/updated)
                                <1>       ;
                                <1>       ; (EAX, EBX, ECX, EDX, EDI will be changed)
                                <1>
0000BA21 29C0                   <1>       sub    eax, eax
0000BA23 8A25[B4890100]         <1>       mov    ah, [Current_Dir_Level]
0000BA29 A0[B5890100]           <1>       mov    al, [Current_FATType]
0000BA2E 3C01                   <1>       cmp    al, 1
0000BA30 723A                   <1>       jb     short loc_UPDLMDT_proc_retn
                                <1>
                                <1> loc_update_parent_dir_lm_date_time:
0000BA32 08E4                   <1>       or     ah, ah
0000BA34 7436                   <1>       jz     short loc_UPDLMDT_proc_retn
                                <1>
0000BA36 56                     <1>       push   esi ; *
0000BA37 8825[54930100]         <1>       mov    [UPDLMDT_CDirLevel], ah
0000BA3D 8B15[B0890100]         <1>       mov    edx, [Current_Dir_FCluster]
0000BA43 8915[55930100]         <1>       mov    [UPDLMDT_CDirFCluster], edx
                                <1>
0000BA49 FECC                   <1>       dec    ah
0000BA4B B90C000000             <1>       mov    ecx, 12
0000BA50 BE[13910100]           <1>         mov     esi, PATH_Array
                                <1>
0000BA55 8825[B4890100]         <1>       mov    [Current_Dir_Level], ah
0000BA5B 08E4                   <1>       or     ah, ah
0000BA5D 750E                   <1>       jnz    short loc_update_parent_dir_lmdt_load_sub_dir_1
0000BA5F 803D[B5890100]02       <1>       cmp    byte [Current_FATType], 2
0000BA66 770B                   <1>       ja     short loc_update_parent_dir_lmdt_load_sub_dir_2
0000BA68 28C0                   <1>       sub    al, al ; eax = 0
0000BA6A EB0A                   <1>       jmp    short loc_update_parent_dir_lmdt_load_sub_dir_3
```

```
                                <1>
                                <1> loc_UPDLMDT_proc_retn:
0000BA6C C3                     <1>         retn
                                <1>
                                <1> loc_update_parent_dir_lmdt_load_sub_dir_1:
0000BA6D B010                   <1>         mov     al, 16
0000BA6F F6E4                   <1>         mul     ah
0000BA71 01C6                   <1>         add     esi, eax
                                <1>
                                <1> loc_update_parent_dir_lmdt_load_sub_dir_2:
0000BA73 8B460C                 <1>         mov     eax, [esi+12] ; Parent Dir First Cluster
                                <1>
                                <1> loc_update_parent_dir_lmdt_load_sub_dir_3:
0000BA76 A3[B0890100]           <1>         mov     [Current_Dir_FCluster], eax
                                <1>
0000BA7B 83C610                 <1>         add     esi, 16
0000BA7E 66BF[3A92]             <1>         mov     di, Dir_File_Name
0000BA82 F3A4                   <1>         rep     movsb
                                <1>
0000BA84 BE00010900             <1>         mov     esi, Logical_DOSDisks
0000BA89 29DB                   <1>         sub     ebx, ebx
0000BA8B 8A3D[B6890100]         <1>         mov     bh, [Current_Drv]
0000BA91 01DE                   <1>         add     esi, ebx
0000BA93 E88EF7FFFF             <1>         call    reload_current_directory
0000BA98 7230                   <1>         jc      short loc_update_parent_dir_lmdt_restore_cdirlevel
                                <1>
                                <1> loc_update_parent_dir_lmdt_locate_dir:
0000BA9A BE[3A920100]           <1>         mov     esi, Dir_File_Name
0000BA9F 6631C9                 <1>         xor     cx, cx
0000BAA2 66B81008               <1>         mov     ax, 0810h ; Only directories
0000BAA6 E8B4F6FFFF             <1>           call    locate_current_dir_file
                                <1>         ; EDI = DirBuff Directory Entry Address
0000BAAB 721D                   <1>         jc      short loc_update_parent_dir_lmdt_restore_cdirlevel
                                <1>
0000BAAD E836FEFFFF             <1>         call    convert_current_date_time
0000BAB2 66895712               <1>         mov     [edi+18], dx ; Last Access Date
0000BAB6 66895718               <1>         mov     [edi+24], dx ; Last Write Date
0000BABA 66894716               <1>         mov     [edi+22], ax ; Last Write Time
                                <1>
0000BABE C605[DC900100]02       <1>         mov     byte [DirBuff_ValidData], 2
0000BAC5 E8BCFEFFFF             <1>         call    save_directory_buffer
                                <1>         ; 29/12/2017
                                <1>         ;jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
                                <1>         ;xor    al, al
                                <1> loc_update_parent_dir_lmdt_restore_cdirlevel:
                                <1>         ;current directory level restoration
0000BACA 8A25[54930100]         <1>         mov     ah, [UPDLMDT_CDirLevel]
0000BAD0 8825[B4890100]         <1>         mov     [Current_Dir_Level], ah
0000BAD6 8B15[55930100]         <1>           mov     edx, [UPDLMDT_CDirFCluster]
0000BADC 8915[B0890100]         <1>         mov     [Current_Dir_FCluster], edx
                                <1>
0000BAE2 5E                     <1>         pop     esi ; *
0000BAE3 C3                     <1>         retn
                                <1>
                                <1> delete_longname:
                                <1>         ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 01/08/2011 (DIR.ASM, 'proc_delete_longname')
                                <1>         ; 14/03/2010
                                <1>         ; INPUT ->
                                <1>         ;       EAX = Directory Entry (Index) Number (< 65536)
                                <1>         ; OUTPUT ->
                                <1>         ;       cf = 0 -> OK  (EAX = 0)
                                <1>         ;       cf = 1 -> error code in EAX (AL)
                                <1>         ;
                                <1>         ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
                                <1>
0000BAE4 66A3[84930100]         <1>         mov     [DLN_EntryNumber], ax
0000BAEA C605[86930100]40       <1>           mov     byte [DLN_40h], 40h
                                <1>
0000BAF1 E858000000             <1>         call    locate_current_dir_entry
0000BAF6 7308                   <1>         jnc     short loc_dln_check_attributes
0000BAF8 C3                     <1>         retn
                                <1>
                                <1> loc_dln_longname_not_found:
0000BAF9 B802000000             <1>         mov     eax, 2
0000BAFE F9                     <1>         stc
0000BAFF C3                     <1>         retn
                                <1>
                                <1> loc_dln_check_attributes:
0000BB00 B00F                   <1>         mov     al, 0Fh  ; long name
0000BB02 8A670B                 <1>         mov     ah, [edi+0Bh] ; dir entry attributes
0000BB05 38C4                   <1>         cmp     ah, al
0000BB07 75F0                   <1>         jne     short loc_dln_longname_not_found
0000BB09 8A27                   <1>         mov     ah, [edi]
0000BB0B 2A25[86930100]         <1>         sub     ah, [DLN_40h]
0000BB11 76E6                   <1>         jna     short loc_dln_longname_not_found
0000BB13 80FC14                 <1>         cmp     ah, 14h ; 84-64=20 -> 20*13=260 bytes
0000BB16 77E1                   <1>         ja      short loc_dln_longname_not_found
                                <1>
0000BB18 C607E5                 <1>         mov     byte [edi], 0E5h  ; deleted sign
0000BB1B C605[DC900100]02       <1>         mov     byte [DirBuff_ValidData], 2 ; changed/write sign
0000BB22 C605[86930100]00       <1>         mov     byte [DLN_40h], 0 ; 40h -> 0
                                <1>
                                <1> loc_dln_delete_next_ln_entry:
0000BB29 80FC01                 <1>         cmp     ah, 1
0000BB2C 7616                   <1>         jna     short loc_dln_longname_retn
                                <1> loc_dln_delete_next_ln_entry_0:
0000BB2E 66FF05[84930100]       <1>         inc     word [DLN_EntryNumber]
0000BB35 0FB705[84930100]       <1>         movzx   eax, word [DLN_EntryNumber]
0000BB3C E80D000000             <1>         call    locate_current_dir_entry
0000BB41 73BD                   <1>         jnc     short loc_dln_check_attributes
                                <1>
                                <1> loc_dln_longname_stc_retn:
0000BB43 C3                     <1>         retn
                                <1>
```

```
                                <1> loc_dln_longname_retn:
                                <1>       ;cmp   byte [DirBuff_ValidData], 2
                                <1>       ;jne   short loc_dln_longname_retn_xor_eax
0000BB44 E83DFEFFFF             <1>       call   save_directory_buffer
0000BB49 72F8                   <1>       jc     short loc_dln_longname_stc_retn
                                <1>
                                <1> loc_dln_longname_retn_xor_eax:
0000BB4B 31C0                   <1>       xor    eax, eax
0000BB4D 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:
0000BB4E 56                     <1>       push   esi
0000BB4F 89C1                   <1>       mov    ecx, eax
0000BB51 BA20000000             <1>       mov    edx, 32
0000BB56 F7E2                   <1>       mul    edx
0000BB58 A3[90930100]           <1>       mov    [LCDE_ByteOffset], eax
0000BB5D 31DB                   <1>       xor    ebx, ebx
0000BB5F 8A3D[B6890100]         <1>       mov    bh, [Current_Drv]
0000BB65 A0[DA900100]           <1>         mov     al, [DirBuff_DRV]
0000BB6A 2C41                   <1>       sub    al, 'A'
0000BB6C BE00010900             <1>         mov     esi, Logical_DOSDisks
0000BB71 01DE                   <1>       add    esi, ebx
0000BB73 38C7                   <1>       cmp    bh, al
0000BB75 0F8592000000           <1>         jne     loc_lcde_reload_current_directory
                                <1> loc_lcde_cdl_check:
0000BB7B 803D[B4890100]00       <1>       cmp    byte [Current_Dir_Level], 0
0000BB82 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)
0000BB84 807E0303               <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
0000BB88 7324                   <1>       jnb    short loc_lcde_calc_dirbuff_cluster_offset
                                <1>
                                <1> loc_lcde_cdl_check_FAT12_16:
0000BB8A 668B4617               <1>       mov    ax, [esi+LD_BPB+RootDirEnts]
0000BB8E 6648                   <1>       dec    ax
                                <1>       ;xor   dx, dx
0000BB90 6639C8                 <1>       cmp    ax, cx ; cx = Directory Entry (Index) Number
0000BB93 720E                   <1>       jb     short loc_lcde_stc_12h_retn
0000BB95 66890D[88930100]       <1>       mov    [LCDE_EntryIndex], cx
0000BB9C 31C0                   <1>       xor    eax, eax
0000BB9E E993000000             <1>         jmp     loc_lcde_check_dir_buffer_cluster
                                <1>
                                <1> loc_lcde_stc_12h_retn:
0000BBA3 5E                     <1>       pop    esi
0000BBA4 89CB                   <1>       mov    ebx, ecx
0000BBA6 89D1                   <1>       mov    ecx, edx
                                <1>       ; 16/10/2016 (12h -> 12)
0000BBA8 B80C000000             <1>       mov    eax, 12 ; No more files
0000BBAD C3                     <1>       retn
                                <1>
                                <1> loc_lcde_calc_dirbuff_cluster_offset:
0000BBAE 8A5E13                 <1>       mov    bl, [esi+LD_BPB+SecPerClust]
0000BBB1 30FF                   <1>       xor    bh, bh
0000BBB3 668B4611               <1>       mov    ax, [esi+LD_BPB+BytesPerSec]
0000BBB7 66F7E3                 <1>       mul    bx
0000BBBA 6609D2                 <1>       or     dx, dx ; If bytes per cluster > 32KB it is invalid
0000BBBD 755D                   <1>       jnz    short loc_lcde_invalid_format
                                <1>       ;mov   ecx, eax
0000BBBF 6689C1                 <1>       mov    cx, ax ; BYTES PER CLUSTER
0000BBC2 A1[90930100]           <1>       mov    eax, [LCDE_ByteOffset]
                                <1>       ;sub   edx, edx
0000BBC7 F7F1                   <1>       div    ecx
0000BBC9 3DFFFF0000             <1>       cmp    eax, 65535
0000BBCE 774C                   <1>       ja     short loc_lcde_invalid_format
                                <1>
                                <1>       ; cluster sequence number of directory (< 65536)
0000BBD0 66A3[8A930100]         <1>       mov    [LCDE_ClusterSN], ax
                                <1>
0000BBD6 6689D0                 <1>       mov    ax, dx ; byte offset in cluster (directory buffer)
0000BBD9 66BB2000               <1>       mov    bx, 32 ; ; 1 dir entry = 32 bytes
0000BBDD 6629D2                 <1>         sub     dx, dx  ; 0
0000BBE0 66F7F3                 <1>       div    bx
0000BBE3 66A3[88930100]         <1>       mov    [LCDE_EntryIndex], ax ; dir entry index/sequence number
                                <1>                                    ; (in directory buffer/cluster)
                                <1> loc_lcde_get_current_sub_dir_fcluster:
0000BBE9 A1[B0890100]           <1>       mov    eax, [Current_Dir_FCluster]
                                <1>
                                <1> loc_lcde_get_next_cluster:
0000BBEE 66833D[8A930100]00     <1>       cmp    word [LCDE_ClusterSN], 0
0000BBF6 763E                   <1>       jna    short loc_lcde_check_dir_buffer_cluster
0000BBF8 A3[8C930100]           <1>       mov    [LCDE_Cluster], eax
```

```
5425 0000BBFD E834100000          <1>       call   get_next_cluster
5426 0000BC02 7220                <1>       jc     short loc_lcde_check_gnc_error
5427 0000BC04 66FF0D[8A930100]    <1>       dec    word [LCDE_ClusterSN]
5428 0000BC0B EBE1                <1>       jmp    short loc_lcde_get_next_cluster
5429                              <1>
5430                              <1> loc_lcde_reload_current_directory:
5431 0000BC0D 51                  <1>       push   ecx
5432 0000BC0E E813F6FFFF          <1>       call   reload_current_directory
5433 0000BC13 59                  <1>       pop    ecx
5434 0000BC14 0F8361FFFFFF        <1>         jnc     loc_lcde_cdl_check
5435 0000BC1A 5E                  <1>       pop    esi
5436 0000BC1B C3                  <1>       retn
5437                              <1>
5438                              <1> loc_lcde_invalid_format:
5439                              <1>       ; 15/10/2016 (0Bh -> 28)
5440 0000BC1C B81C000000          <1>       mov    eax, 28 ; Invalid Format !
5441                              <1> loc_lcde_drive_not_ready_read_err:
5442 0000BC21 F9                  <1>       stc
5443 0000BC22 5E                  <1>       pop    esi
5444 0000BC23 C3                  <1>       retn
5445                              <1>
5446                              <1> loc_lcde_check_gnc_error:
5447 0000BC24 09C0                <1>       or     eax, eax
5448 0000BC26 75F9                <1>       jnz    short loc_lcde_drive_not_ready_read_err
5449 0000BC28 66FF0D[8A930100]    <1>       dec    word [LCDE_ClusterSN]
5450 0000BC2F 75EB                <1>       jnz    short loc_lcde_invalid_format
5451 0000BC31 A1[8C930100]        <1>       mov    eax, [LCDE_Cluster]
5452                              <1>
5453                              <1> loc_lcde_check_dir_buffer_cluster:
5454 0000BC36 3B05[E1900100]      <1>       cmp    eax, [DirBuff_Cluster]
5455 0000BC3C 755C                <1>       jne    short loc_lcde_load_dir_cluster
5456 0000BC3E 803D[DC900100]00    <1>       cmp    byte [DirBuff_ValidData], 0
5457 0000BC45 7727                <1>       ja     short lcde_check_dir_buffer_cluster_next
5458 0000BC47 803D[B4890100]00    <1>       cmp    byte [Current_Dir_Level], 0
5459 0000BC4E 775F                <1>       ja     short loc_lcde_load_dir_cluster_0
5460                              <1>       ; 27/02/2016
5461                              <1>       ; TRDOS v1 has bug here for FAT32 fs !
5462 0000BC50 807E0303            <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
5463 0000BC54 7359                <1>       jnb    short loc_lcde_load_dir_cluster_0
5464                              <1>       ;
5465 0000BC56 0FB74E17            <1>       movzx  ecx, word [esi+LD_BPB+RootDirEnts]
5466 0000BC5A 6683C10F            <1>       add    cx, 15 ; round up (16 entries per sector)
5467 0000BC5E 66C1E904            <1>       shr    cx, 4 ; 1 sector contains 16 dir entries
5468                              <1>
5469 0000BC62 8B4664              <1>         mov     eax, [esi+LD_ROOTBegin]
5470 0000BC65 EB54                <1>       jmp    short loc_lcde_load_dir_cluster_1
5471                              <1>
5472                              <1> loc_lcde_validate_dirBuff:
5473 0000BC67 C605[DC900100]01    <1>       mov    byte [DirBuff_ValidData], 1
5474                              <1>
5475                              <1> lcde_check_dir_buffer_cluster_next:
5476 0000BC6E 0FB71D[88930100]    <1>       movzx  ebx, word [LCDE_EntryIndex]
5477 0000BC75 663B1D[DF900100]    <1>       cmp    bx, [DirBuff_LastEntry]
5478 0000BC7C 779E                <1>       ja     short loc_lcde_invalid_format
5479 0000BC7E B820000000          <1>       mov    eax, 32
5480 0000BC83 F7E3                <1>       mul    ebx
5481                              <1>       ;or    edx, edx
5482                              <1>       ;jnz   short loc_lcde_invalid_format
5483                              <1>
5484 0000BC85 BF00000800          <1>       mov    edi, Directory_Buffer
5485 0000BC8A 01C7                <1>       add    edi, eax ; add entry offset to buffer address
5486                              <1>
5487                              <1> loc_lcde_dir_buffer_last_check:
5488 0000BC8C A1[E1900100]        <1>       mov    eax, [DirBuff_Cluster]
5489 0000BC91 0FB60D[DC900100]    <1>       movzx  ecx, byte [DirBuff_ValidData]
5490                              <1>
5491                              <1> loc_lcde_retn:
5492 0000BC98 5E                  <1>       pop    esi
5493 0000BC99 C3                  <1>       retn
5494                              <1>
5495                              <1> loc_lcde_load_dir_cluster:
5496                              <1>       ;cmp   byte [DirBuff_ValidData], 2
5497                              <1>       ;jne   short loc_lcde_load_dir_cluster_n2
5498 0000BC9A 50                  <1>       push   eax
5499 0000BC9B E8E6FCFFFF          <1>       call   save_directory_buffer
5500 0000BCA0 58                  <1>       pop    eax
5501 0000BCA1 72F5                <1>       jc     short loc_lcde_retn
5502                              <1>
5503                              <1> loc_lcde_load_dir_cluster_n2:
5504 0000BCA3 C605[DC900100]00    <1>       mov    byte [DirBuff_ValidData], 0
5505 0000BCAA A3[E1900100]        <1>       mov    [DirBuff_Cluster], eax
5506                              <1>
5507                              <1> loc_lcde_load_dir_cluster_0:
5508 0000BCAF 83E802              <1>       sub    eax, 2
5509 0000BCB2 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
5510 0000BCB6 F7E1                <1>       mul    ecx
5511 0000BCB8 034668              <1>         add     eax, [esi+LD_DATABegin]
5512                              <1>
5513                              <1> loc_lcde_load_dir_cluster_1:
5514 0000BCBB BB00000800          <1>       mov    ebx, Directory_Buffer
5515                              <1>       ; ecx = sector count
5516 0000BCC0 E8B86D0000          <1>       call   disk_read
5517 0000BCC5 73A0                <1>       jnc    short loc_lcde_validate_dirBuff
5518                              <1>
5519                              <1>       ; 15/10/2016
5520                              <1>       ; (Disk read error instead of drv not ready err)
5521 0000BCC7 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error !
5522 0000BCCC EBCA                <1>       jmp    short loc_lcde_retn
5523                              <1>
5524                              <1>
5525                              <1> remove_file:
5526                              <1>       ; 15/10/2016
5527                              <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
5528                              <1>       ; 10/04/2011 (FILE.ASM, 'proc_delete_file')
5529                              <1>       ; 09/08/2010
```

```
5530                            <1>       ; INPUT ->
5531                            <1>       ;      EDI = Directory Buffer Entry Address
5532                            <1>       ;       CX = Directory Buffer Entry Counter/Index
5533                            <1>       ;       BL = Longname Entry Length
5534                            <1>       ;       BH = Logical DOS Drive Number
5535                            <1>
5536 0000BCCE 29C0             <1>       sub    eax, eax
5537 0000BCD0 88FC             <1>       mov    ah, bh
5538 0000BCD2 BE00010900       <1>       mov    esi, Logical_DOSDisks
5539 0000BCD7 01C6             <1>       add    esi, eax
5540                            <1>
5541 0000BCD9 807E0301         <1>       cmp    byte [esi+LD_FATType], 1
5542 0000BCDD 7312             <1>       jnb    short loc_del_fat_file
5543                            <1>
5544 0000BCDF 807E04A1         <1>       cmp    byte [esi+LD_FSType], 0A1h
5545 0000BCE3 7406             <1>       je     short loc_del_fs_file
5546                            <1>
5547                            <1> loc_del_file_invalid_format:
5548 0000BCE5 30E4             <1>       xor    ah, ah
5549                            <1>       ; 15/10/2016 (0Bh -> 28)
5550 0000BCE7 B01C             <1>       mov    al, 28  ; Invalid Format
5551 0000BCE9 F9               <1>       stc
5552 0000BCEA C3               <1>       retn
5553                            <1>
5554                            <1> loc_del_fs_file:
5555 0000BCEB E83F0F0000       <1>       call   delete_fs_file
5556 0000BCF0 C3               <1>       retn
5557                            <1>
5558                            <1> loc_del_fat_file:
5559 0000BCF1 E808000000       <1>       call   delete_directory_entry
5560 0000BCF6 7205             <1>       jc     short loc_del_file_err_retn
5561                            <1>
5562                            <1> loc_delfile_unlink_cluster_chain:
5563 0000BCF8 E863170000       <1>       call   truncate_cluster_chain
5564                            <1>       ;jc    short loc_del_file_err_retn
5565                            <1>
5566                            <1> loc_delfile_return:
5567                            <1> loc_del_file_err_retn:
5568 0000BCFD C3               <1>       retn
5569                            <1>
5570                            <1> delete_directory_entry:
5571                            <1>       ; 15/10/2016
5572                            <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
5573                            <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_directory_entry')
5574                            <1>       ; 10/04/2011
5575                            <1>       ; INPUT ->
5576                            <1>       ;      ESI = Logical Dos Drive Descripton Table Address
5577                            <1>       ;      EDI = Directory Buffer Entry Address
5578                            <1>       ;       CX = Directory Buffer Entry Counter/Index
5579                            <1>       ;       BL = Longname Entry Length
5580                            <1>       ; OUTPUT ->
5581                            <1>       ;      ESI = Logical dos drive descripton table address
5582                            <1>       ;      EAX = First cluster to be truncated/unlinked
5583                            <1>       ;       CF = 1 -> Error code in EAX (AL)
5584                            <1>       ;       CF = 0 & BH <> 0 -> LMDT write error  (BH = 1)
5585                            <1>       ;       CF = 0 & BL <> 0 -> Long name delete error (BL = FFh)
5586                            <1>       ;
5587                            <1>       ;  (EDI, EBX, ECX register contents will be changed)
5588                            <1>
5589 0000BCFE 881D[1E930100]   <1>       mov    [DelFile_LNEL], bl
5590 0000BD04 66890D[1C930100] <1>       mov    [DelFile_EntryCounter], cx
5591                            <1>
5592 0000BD0B 668B4714         <1>       mov    ax, [edi+20] ; First Cluster High Word
5593 0000BD0F C1E010           <1>       shl    eax, 16
5594 0000BD12 668B471A         <1>       mov    ax, [edi+26] ; First Cluster Low Word
5595                            <1>
5596 0000BD16 A3[18930100]     <1>       mov    [DelFile_FCluster], eax
5597                            <1>
5598                            <1> loc_del_short_name:
5599 0000BD1B C607E5           <1>       mov    byte [edi], 0E5h  ; Deleted sign
5600                            <1>
5601 0000BD1E C605[DC900100]02 <1>       mov    byte [DirBuff_ValidData], 2
5602 0000BD25 E85CFCFFFF       <1>       call   save_directory_buffer
5603 0000BD2A 723D             <1>       jc     short loc_delete_direntry_err_return
5604                            <1>
5605                            <1> loc_del_long_name:
5606 0000BD2C 0FB615[1E930100] <1>       movzx  edx, byte [DelFile_LNEL]
5607 0000BD33 08D2             <1>       or     dl, dl
5608 0000BD35 7416             <1>       jz     short loc_del_dir_entry_update_parent_dir_lm_date
5609                            <1>
5610 0000BD37 8835[1E930100]   <1>       mov    byte [DelFile_LNEL], dh ; 0
5611                            <1>
5612 0000BD3D 0FB705[1C930100] <1>       movzx  eax,  word [DelFile_EntryCounter]
5613 0000BD44 29D0             <1>       sub    eax, edx
5614                            <1>       ;jnc   short loc_del_long_name_continue
5615 0000BD46 7205             <1>       jc     short loc_del_dir_entry_update_parent_dir_lm_date
5616                            <1>
5617                            <1> ;loc_del_direntry_inv_data_return: ; 15/10/2016 (0Dh -> 29)
5618                            <1> ;     mov    eax, 29 ; 0Dh (TRDOS 8086) ; Invalid data
5619                            <1> ;     retn
5620                            <1>
5621                            <1> loc_del_long_name_continue:
5622                            <1>       ; AX = Directory Entry Number of the long name last entry
5623 0000BD48 E897FDFFFF       <1>       call   delete_longname
5624                            <1>       ;jc    short loc_delete_direntry_err_return
5625                            <1>
5626                            <1> loc_del_dir_entry_update_parent_dir_lm_date:
5627 0000BD4D 801D[1E930100]00 <1>       sbb    byte [DelFile_LNEL], 0 ; 0FFh if cf = 1
5628                            <1>
5629 0000BD54 E8C8FCFFFF       <1>       call   update_parent_dir_lmdt
5630 0000BD59 B700             <1>       mov    bh, 0
5631 0000BD5B 80D700           <1>       adc    bh, 0
5632                            <1>
5633 0000BD5E 8A1D[1E930100]   <1>       mov    bl, byte [DelFile_LNEL]
5634                            <1>
```

```
                            <1> loc_delete_direntry_return:
0000BD64 A1[18930100]       <1>       mov    eax, [DelFile_FCluster]
                            <1> loc_delete_direntry_err_return:
0000BD69 C3                 <1>       retn
                            <1>
                            <1> rename_directory_entry:
                            <1>       ; 13/11/2017
                            <1>       ; 15/10/2016
                            <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 01/08/2011 (DIR.ASM, 'proc_rename_directory_entry')
                            <1>       ; 19/11/2010
                            <1>       ; INPUT -> (Current Directory)
                            <1>       ;      CX = Directory Entry Number
                            <1>       ;       EAX = First Cluster number of file or directory
                            <1>       ;       EBX = Longname Length (dir entry count) (< 256)
                            <1>       ;       ESI = New file (or directory) name (no path).
                            <1>       ;            (ASCIIZ string)
                            <1>       ; OUTPUT ->
                            <1>       ;       CF = 0 -> successfull
                            <1>       ;       CF = 1 -> error code in EAX (AL)
                            <1>       ;
                            <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                            <1>
0000BD6A 803D[B5890100]00   <1>       cmp    byte [Current_FATType], 0
0000BD71 7706               <1>       ja     short loc_rename_directory_entry
                            <1>
0000BD73 E8B80E0000         <1>       call   rename_fs_file_or_directory
0000BD78 C3                 <1>       retn
                            <1>
                            <1> loc_rename_directory_entry:
0000BD79 881D[1E930100]     <1>       mov    [DelFile_LNEL], bl
0000BD7F 66890D[1C930100]   <1>       mov    [DelFile_EntryCounter], cx
0000BD86 A3[18930100]       <1>       mov    [DelFile_FCluster], eax
                            <1>
0000BD8B 0FB7C1             <1>       movzx  eax, cx
0000BD8E E8BBFDFFFF         <1>       call   locate_current_dir_entry
0000BD93 7308               <1>       jnc    short loc_rename_direntry_check_fcluster
                            <1>
                            <1> loc_rename_direntry_pop_retn:
0000BD95 C3                 <1>       retn
                            <1>
                            <1> loc_rename_direntry_pop_invd_retn:
0000BD96 F9                 <1>       stc
                            <1> loc_rename_direntry_invd_retn:
                            <1>       ; 15/10/2016 (0Dh -> 29)
0000BD97 B81D000000         <1>       mov    eax, 29 ; Invalid data
                            <1> loc_rename_retn:
0000BD9C C3                 <1>       retn
                            <1>
                            <1> loc_rename_direntry_check_fcluster:
0000BD9D 668B5714           <1>       mov    dx, [edi+20] ; First Cluster HW
0000BDA1 C1E210             <1>       shl    edx, 16 ; 13/11/2017
0000BDA4 668B571A           <1>       mov    dx, [edi+26] ; First Cluster LW
0000BDA8 3B15[18930100]     <1>       cmp    edx, [DelFile_FCluster]
0000BDAE 75E6               <1>       jne    short loc_rename_direntry_pop_invd_retn
                            <1>       ; ESI = New file (or directory) name. (ASCIIZ string)
                            <1>       ; 06/03/2016
                            <1>       ; TRDOS v2 - NOTE: 'convert_file_name' procedure
                            <1>       ; has been modified for eliminating following situation.
                            <1>       ;
                            <1>       ; TRDOS v1 - NOTE: If file/dir name is more than 11 bytes
                            <1>       ; without a dot, attributes (edi+11) byte will be overwritten !
                            <1>       ; (Dot file name input must be proper for 11 byte dir entry
                            <1>       ;  type file name output.)
0000BDB0 E8A2F6FFFF         <1>       call   convert_file_name
                            <1>
0000BDB5 C605[DC900100]02   <1>         mov     byte [DirBuff_ValidData], 2
0000BDBC E8C5FBFFFF         <1>       call   save_directory_buffer
0000BDC1 72D9               <1>       jc     short loc_rename_retn
                            <1>
                            <1> loc_rename_direntry_del_ln:
0000BDC3 0FB615[1E930100]   <1>       movzx  edx, byte [DelFile_LNEL]
0000BDCA 08D2               <1>       or     dl, dl
0000BDCC 7410               <1>       jz     short loc_rename_direntry_update_parent_dir_lm_date
                            <1>
0000BDCE 0FB705[1C930100]   <1>       movzx  eax, word [DelFile_EntryCounter]
0000BDD5 29D0               <1>       sub    eax, edx
0000BDD7 72BE               <1>       jc     short loc_rename_direntry_invd_retn
                            <1>
                            <1> loc_rename_direntry_del_ln_continue:
                            <1>       ; EAX = Directory Entry Number of the long name last entry
0000BDD9 E806FDFFFF         <1>       call   delete_longname
                            <1>
                            <1> loc_rename_direntry_update_parent_dir_lm_date:
0000BDDE E83EFCFFFF         <1>       call   update_parent_dir_lmdt
0000BDE3 31C0               <1>       xor    eax, eax
0000BDE5 C3                 <1>       retn
                            <1>
                            <1> move_source_file_to_destination_file:
                            <1>       ; 15/10/2016
                            <1>       ; 11/03/2016
                            <1>       ; 10/03/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 01/08/2011 (FILE.ASM)
                            <1>       ; 04/08/2010
                            <1>       ;
                            <1>       ;   Phase 1 -> Check destination file,
                            <1>       ;              'not found' is required
                            <1>       ;   Phase 2 -> Check source file
                            <1>       ;              'found' and proper attributes is required
                            <1>       ;   Phase 3 -> Make destination directory entry,
                            <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>
0000BDE6 3C02           <1>       cmp    al, 2
0000BDE8 0F847F010000   <1>       je     msftdf_df2_check_directory
0000BDEE A2[9E940100]   <1>       mov    [move_cmd_phase], al
                        <1>
                        <1> msftdf_parse_sf_path:
                        <1>       ; ESI = ASCIIZ pathname (Source)
0000BDF3 57             <1>       push   edi
0000BDF4 BF[9C930100]   <1>       mov    edi, SourceFile_Drv
0000BDF9 E824F7FFFF     <1>       call   parse_path_name
0000BDFE 5E             <1>       pop    esi
0000BDFF 7211           <1>       jc     short msftdf_psf_retn
                        <1>
                        <1> msftdf_parse_df_path:
                        <1>       ; ESI = ASCIIZ pathname   (Destination)
0000BE01 BF[1C940100]   <1>       mov    edi, DestinationFile_Drv
0000BE06 E817F7FFFF     <1>       call   parse_path_name
0000BE0B 7306           <1>       jnc    short msftdf_check_sf_drv
                        <1>
0000BE0D 3C01           <1>       cmp    al, 1 ; File or directory name is not existing
0000BE0F 7602           <1>       jna    short msftdf_check_sf_drv
                        <1>
                        <1> msftdf_stc_retn:
0000BE11 F9             <1>       stc
                        <1> msftdf_psf_retn:
0000BE12 C3             <1>       retn
                        <1>
                        <1> msftdf_check_sf_drv:
0000BE13 A0[9C930100]   <1>       mov    al, [SourceFile_Drv]
                        <1>
                        <1> msftdf_check_df_drv:
0000BE18 8A15[1C940100] <1>       mov    dl, [DestinationFile_Drv]
                        <1>
                        <1> msftdf_compare_sf_df_drv:
0000BE1E 29DB           <1>       sub    ebx, ebx
0000BE20 8A3D[B6890100] <1>       mov    bh, [Current_Drv]
0000BE26 38C2           <1>       cmp    dl, al
0000BE28 7409           <1>       je     short msftdf_check_sf_df_drv_ok
                        <1>
                        <1> msftdf_not_same_drv:
                        <1>         ; DL = source file's drive number
0000BE2A 88C6           <1>       mov    dh, al ; destination file's drive number
                        <1>       ; 15/10/2016 (11h -> 21)
0000BE2C B815000000     <1>       mov    eax, 21 ; Not the same drive
0000BE31 F9             <1>       stc
0000BE32 C3             <1>       retn
                        <1>
                        <1> msftdf_check_sf_df_drv_ok:
0000BE33 8815[9F940100] <1>       mov    [msftdf_sf_df_drv], dl
                        <1>
0000BE39 29C0           <1>         sub  eax, eax
0000BE3B 88D4           <1>       mov    ah, dl
0000BE3D 0500010900     <1>       add    eax, Logical_DOSDisks
0000BE42 A3[A0940100]   <1>       mov    [msftdf_drv_offset], eax
                        <1>
0000BE47 38FA           <1>       cmp    dl, bh ; byte [Current_Drv]
0000BE49 7407           <1>       je     short msftdf_df_check_directory
                        <1>
                        <1> msftdf_change_drv:
0000BE4B E85EC1FFFF     <1>       call   change_current_drive
0000BE50 726D           <1>       jc     short msftdf_df_error_retn
                        <1>
                        <1> msftdf_check_destination_file:
                        <1> msftdf_df_check_directory:
0000BE52 BE[1D940100]   <1>       mov    esi, DestinationFile_Directory
0000BE57 803E20         <1>       cmp    byte [esi], 20h
0000BE5A 760F           <1>       jna    short msftdf_df_find_1
                        <1>
                        <1> msftdf_df_change_directory:
0000BE5C FE05[833F0100] <1>       inc    byte [Restore_CDIR]
0000BE62 30E4           <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000BE64 E8A3F0FFFF     <1>       call   change_current_directory
0000BE69 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:
0000BE6B BE[5E940100]   <1>         mov     esi, DestinationFile_Name
0000BE70 803E20         <1>       cmp    byte [esi], 20h
0000BE73 7631           <1>       jna    short msftdf_df_copy_sf_name
                        <1>
                        <1> msftdf_df_find_2:
0000BE75 6631C0         <1>       xor    ax, ax ; DestinationFile_AttributesMask -> any/zero
```

```
5845 0000BE78 E8D4D4FFFF          <1>       call   find_first_file
5846 0000BE7D 0F838D000000        <1>       jnc    msftdf_permission_denied_retn
5847                              <1>
5848                              <1> msftdf_df_check_error_code:
5849                              <1>       ;cmp   eax, 2 ; File not found error
5850 0000BE83 3C02                <1>       cmp    al, 2
5851 0000BE85 7537                <1>       jne    short msftdf_df_stc_retn
5852                              <1>
5853                              <1> msftdf_df_check_fname:
5854                              <1>       ; 15/10/2016
5855 0000BE87 BE[5E940100]        <1>       mov    esi, DestinationFile_Name ; *
5856 0000BE8C E87ED8FFFF          <1>       call   check_filename
5857 0000BE91 7307                <1>       jnc    short msftdf_convert_df_direntry_name
5858                              <1>       ; invalid file name chars !
5859 0000BE93 B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
5860 0000BE98 EB24                <1>       jmp    short msftdf_df_stc_retn
5861                              <1>
5862                              <1> msftdf_convert_df_direntry_name:
5863                              <1>       ; mov  esi, DestinationFile_Name ; *
5864 0000BE9A BF[6E940100]        <1>       mov    edi, DestinationFile_DirEntry
5865 0000BE9F E8B3F5FFFF          <1>       call   convert_file_name
5866 0000BEA4 EB1A                <1>       jmp    short msftdf_restore_current_dir_1
5867                              <1>
5868                              <1> msftdf_df_copy_sf_name:
5869 0000BEA6 89F7                <1>       mov    edi, esi
5870 0000BEA8 57                  <1>       push   edi
5871 0000BEA9 BE[DE930100]        <1>         mov     esi, SourceFile_Name
5872 0000BEAE B90C000000          <1>       mov    ecx, 12
5873                              <1> msftdf_df_copy_sf_name_loop:
5874 0000BEB3 AC                  <1>       lodsb
5875 0000BEB4 AA                  <1>         stosb
5876 0000BEB5 08C0                <1>       or     al, al
5877 0000BEB7 7402                <1>       jz     short msftdf_df_copy_sf_name_ok
5878 0000BEB9 E2F8                <1>         loop    msftdf_df_copy_sf_name_loop
5879                              <1> msftdf_df_copy_sf_name_ok:
5880 0000BEBB 5E                  <1>       pop    esi
5881 0000BEBC EBB7                <1>       jmp    short msftdf_df_find_2
5882                              <1>
5883                              <1> msftdf_df_stc_retn:
5884 0000BEBE F9                  <1>       stc
5885                              <1> msftdf_restore_cdir_failed:
5886                              <1> msftdf_df_error_retn:
5887 0000BEBF C3                  <1>       retn
5888                              <1>
5889                              <1> msftdf_restore_current_dir_1:
5890 0000BEC0 803D[833F0100]00    <1>       cmp    byte [Restore_CDIR], 0
5891 0000BEC7 760D                <1>       jna    short msftdf_sf_check_directory
5892 0000BEC9 8B35[A0940100]      <1>       mov    esi, [msftdf_drv_offset]
5893 0000BECF E891C1FFFF          <1>       call   restore_current_directory
5894 0000BED4 72E9                <1>       jc     short msftdf_restore_cdir_failed
5895                              <1>
5896                              <1> msftdf_sf_check_directory:
5897 0000BED6 BE[9D930100]        <1>       mov    esi, SourceFile_Directory
5898 0000BEDB 803E20              <1>       cmp    byte [esi], 20h
5899 0000BEDE 760F                <1>       jna    short msftdf_sf_find
5900                              <1> msftdf_sf_change_directory:
5901 0000BEE0 FE05[833F0100]      <1>       inc    byte [Restore_CDIR]
5902 0000BEE6 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
5903 0000BEE8 E81FF0FFFF          <1>       call   change_current_directory
5904 0000BEED 7227                <1>       jc     short msftdf_return
5905                              <1>
5906                              <1> ;msftdf_sf_change_prompt_dir_string:
5907                              <1> ;     call   change_prompt_dir_string
5908                              <1>
5909                              <1> msftdf_sf_find:
5910 0000BEEF BE[DE930100]        <1>         mov     esi, SourceFile_Name  ; Offset 66
5911 0000BEF4 66B80018            <1>       mov    ax, 1800h ; Only files
5912 0000BEF8 E854D4FFFF          <1>       call   find_first_file
5913 0000BEFD 7217                <1>       jc     short msftdf_return
5914                              <1>
5915                              <1> msftdf_sf_ambgfn_check:
5916 0000BEFF 6609D2              <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
5917 0000BF02 7407                <1>       jz     short msftdf_sf_found
5918                              <1>
5919                              <1> msftdf_ambiguous_file_name_error:
5920 0000BF04 B802000000          <1>       mov    eax, 2 ; File not found error
5921 0000BF09 F9                  <1>       stc
5922 0000BF0A C3                  <1>       retn
5923                              <1>
5924                              <1> msftdf_sf_found:
5925 0000BF0B 80E31F              <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
5926 0000BF0E 7416                <1>       jz     short msftdf_save_sf_structure
5927                              <1>
5928                              <1> msftdf_permission_denied_retn:
5929 0000BF10 B805000000          <1>       mov    eax, 05h ; Access (Permission) denied !
5930 0000BF15 F9                  <1>       stc
5931                              <1> msftdf_rest_cdir_err_retn:
5932                              <1> msftdf_return:
5933 0000BF16 C3                  <1>       retn
5934                              <1>
5935                              <1> msftdf_phase_1_return:
5936 0000BF17 31C0                <1>       xor    eax, eax
5937 0000BF19 A2[9E940100]        <1>       mov    [move_cmd_phase], al ; 0
5938 0000BF1E FEC0                <1>       inc    al ; mov al, 1
5939 0000BF20 BB[6DBF0000]        <1>       mov    ebx, msftdf_df2_check_directory
5940                              <1>       ;mov   edx, 0FFFFFFFFh
5941 0000BF25 C3                  <1>       retn
5942                              <1>
5943                              <1> msftdf_save_sf_structure:
5944 0000BF26 BE[A8920100]        <1>       mov    esi, FindFile_DirEntry
5945 0000BF2B BF[EE930100]        <1>       mov    edi, SourceFile_DirEntry
5946 0000BF30 B908000000          <1>       mov    ecx, 8
5947 0000BF35 F3A5                <1>       rep    movsd
5948                              <1>
5949                              <1> msftdf_df_copy_sf_parameters:
```

```
0000BF37 BE0B000000          <1>       mov    esi, 11
0000BF3C 89F7                <1>       mov    edi, esi
0000BF3E 81C6[EE930100]      <1>       add    esi, SourceFile_DirEntry
0000BF44 81C7[6E940100]      <1>       add    edi, DestinationFile_DirEntry
                             <1>       ;mov   ecx, 21
0000BF4A B115                <1>       mov    cl, 21
0000BF4C F3A4                <1>       rep    movsb
                             <1>
                             <1> msftdf_restore_current_dir_2:
0000BF4E 803D[833F0100]00    <1>       cmp    byte [Restore_CDIR], 0
0000BF55 760D                <1>       jna    short msftdf_df2_check_move_cmd_phase
0000BF57 8B35[A0940100]      <1>       mov    esi, [msftdf_drv_offset]
0000BF5D E803C1FFFF          <1>       call   restore_current_directory
0000BF62 72B2                <1>       jc     short msftdf_rest_cdir_err_retn
                             <1>
                             <1> msftdf_df2_check_move_cmd_phase:
0000BF64 803D[9E940100]01    <1>       cmp    byte [move_cmd_phase], 1
0000BF6B 74AA                <1>       je     short msftdf_phase_1_return
                             <1>
                             <1> msftdf_df2_check_directory:
0000BF6D BE[1D940100]        <1>       mov    esi, DestinationFile_Directory
0000BF72 803E20              <1>       cmp    byte [esi], 20h
0000BF75 760F                <1>       jna    short msftdf_make_dfde_locate_ffe_on_directory
                             <1> msftdf_df2_change_directory:
0000BF77 FE05[833F0100]      <1>       inc    byte [Restore_CDIR]
0000BF7D 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000BF7F E888EFFFFF          <1>       call   change_current_directory
0000BF84 7290                <1>       jc     short msftdf_return
                             <1>
                             <1> ;msftdf_df2_change_prompt_dir_string:
                             <1> ;     call   change_prompt_dir_string
                             <1>
                             <1> msftdf_make_dfde_locate_ffe_on_directory:
                             <1>       ; Current directory fcluster <> Directory buffer cluster
                             <1>       ; Current directory will be reloaded by
                             <1>       ; 'locate_current_dir_file' procedure
                             <1>       ;
                             <1>       ;xor   ax, ax
0000BF86 31C0                <1>       xor    eax, eax
0000BF88 89C1                <1>       mov    ecx, eax
0000BF8A 6649                <1>       dec    cx ; FFFFh
                             <1>              ; CX = FFFFh -> find first deleted or free entry
                             <1>              ; ESI would be ASCIIZ filename address if the call
                             <1>              ; would not be for first free or deleted dir entry
0000BF8C E8CEF1FFFF          <1>       call   locate_current_dir_file
0000BF91 733F                <1>       jnc    msftdf_make_dfde_set_ff_dir_entry
                             <1>
                             <1>       ;cmp   eax, 2
0000BF93 3C02                <1>         cmp  al, 2
0000BF95 7537                <1>       jne    short msftdf_error_retn
                             <1>
                             <1> msftdf_add_new_dir_entry_check_fs:
0000BF97 8B35[A0940100]      <1>       mov    esi, [msftdf_drv_offset]
0000BF9D A1[E1900100]        <1>       mov    eax, [DirBuff_Cluster]
0000BFA2 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
0000BFA6 7711                <1>       ja     short msftdf_add_new_subdir_cluster
                             <1>
                             <1> msftdf_add_new_fs_subdir_section:
                             <1>       ;CL=0, CH=E5h --> deleted entry, CH=0 --> free entry
                             <1>         ;xor cx, cx
0000BFA8 30ED                <1>       xor    ch, ch ; cx = 0 --> add a new subdir section
0000BFAA E8830C0000          <1>       call   add_new_fs_section
0000BFAF 721E                <1>         jc   short msftdf_dsfde_error_retn
                             <1>       ;mov   [createfile_LastDirCluster], eax
                             <1>
0000BFB1 E8A30E0000          <1>       call   load_FS_sub_directory
                             <1>       ;mov   ebx, Directory_Buffer
0000BFB6 7318                <1>       jnc    short msftdf_add_new_fs_subdir_section_ok
0000BFB8 C3                  <1>       retn
                             <1>
                             <1> msftdf_add_new_subdir_cluster:
0000BFB9 E881150000          <1>       call   add_new_cluster
0000BFBE 720F                <1>       jc     short msftdf_dsfde_error_retn
                             <1>
                             <1>       ;mov   [createfile_LastDirCluster], eax
                             <1>
0000BFC0 E8570E0000          <1>       call   load_FAT_sub_directory
0000BFC5 7309                <1>       jnc    short msftdf_add_new_subdir_cluster_ok
                             <1>       ; EBX = Directory buffer address
                             <1>
                             <1> msftdf_ansdc_update_parent_dir_lmdt:
                             <1> msftdf_make_dfde_err_upd_pdir_lmdt:
0000BFC7 50                  <1>       push   eax
0000BFC8 E854FAFFFF          <1>       call   update_parent_dir_lmdt
0000BFCD 58                  <1>       pop    eax
                             <1>
                             <1> msftdf_error_retn:
0000BFCE F9                  <1>       stc
                             <1> msftdf_dsfde_restore_cdir_failed:
                             <1> msftdf_dsfde_error_retn:
0000BFCF C3                  <1>       retn
                             <1>
                             <1> msftdf_add_new_fs_subdir_section_ok:
                             <1> msftdf_add_new_subdir_cluster_ok:
0000BFD0 89DF                <1>       mov    edi, ebx ; Directory buffer address
                             <1>
                             <1> msftdf_make_dfde_set_ff_dir_entry:
0000BFD2 8B15[B0890100]      <1>       mov    edx, [Current_Dir_FCluster]
0000BFD8 8915[04950100]      <1>       mov    [createfile_FFCluster], edx
                             <1>       ; EDI = Directory entry offset
0000BFDE BE[6E940100]        <1>       mov    esi, DestinationFile_DirEntry
0000BFE3 B908000000          <1>       mov    ecx, 8
0000BFE8 F3A5                <1>       rep    movsd
                             <1>
0000BFEA C605[DC900100]02    <1>       mov    byte [DirBuff_ValidData], 2
```

```
6055 0000BFF1 E890F9FFFF          <1>       call   save_directory_buffer
6056 0000BFF6 72CF                <1>       jc     short msftdf_make_dfde_err_upd_pdir_lmdt
6057                              <1>
6058                              <1> msftdf_make_dfde_update_pdir_lmdt:
6059 0000BFF8 E824FAFFFF          <1>       call   update_parent_dir_lmdt
6060                              <1>
6061                              <1> msftdf_dsfde_restore_current_dir_1:
6062 0000BFFD 803D[833F0100]00    <1>       cmp    byte [Restore_CDIR], 0
6063 0000C004 760D                <1>       jna    short msftdf_dsfde_check_directory
6064 0000C006 8B35[A0940100]      <1>       mov    esi, [msftdf_drv_offset]
6065 0000C00C E854C0FFFF          <1>       call   restore_current_directory
6066 0000C011 72BC                <1>       jc     short msftdf_dsfde_restore_cdir_failed
6067                              <1>
6068                              <1> msftdf_dsfde_check_directory:
6069 0000C013 BE[9D930100]        <1>       mov    esi, SourceFile_Directory
6070 0000C018 803E20              <1>       cmp    byte [esi], 20h
6071 0000C01B 760F                <1>       jna    short msftdf_dsfde_find_file
6072                              <1>
6073                              <1> msftdf_dsfde_change_directory:
6074 0000C01D FE05[833F0100]      <1>       inc    byte [Restore_CDIR]
6075 0000C023 28E4                <1>       sub    ah, ah ; CD_COMMAND sign -> 0
6076 0000C025 E8E2EEFFFF          <1>       call   change_current_directory
6077 0000C02A 72A3                <1>       jc     short msftdf_dsfde_error_retn
6078                              <1>
6079                              <1> ;msftdf_dsfde_sf_change_prompt_dir_string:
6080                              <1> ;     call   change_prompt_dir_string
6081                              <1>
6082                              <1> msftdf_dsfde_find_file:
6083 0000C02C BE[DE930100]        <1>       mov    esi, SourceFile_Name  ; Offset 66
6084 0000C031 668B460E            <1>       mov    ax, [esi+14] ; 80 -> SourceFile_AttributesMask
6085 0000C035 E817D3FFFF          <1>       call   find_first_file
6086 0000C03A 7293                <1>       jc     short msftdf_dsfde_error_retn
6087                              <1>
6088                              <1> msftdf_dsfde_delete_direntry:
6089 0000C03C 8B35[A0940100]      <1>       mov    esi, [msftdf_drv_offset]
6090                              <1>
6091 0000C042 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
6092 0000C046 770A                <1>       ja     short msftdf_delete_FAT_direntry
6093                              <1>
6094 0000C048 30DB                <1>       xor    bl, bl
6095                              <1>       ; BL = 0 -> File
6096                              <1>       ; EDI -> Directory buffer entry offset/address
6097 0000C04A E8E40B0000          <1>       call   delete_fs_directory_entry
6098 0000C04F 7315                <1>       jnc    short msftdf_dsfde_restore_current_dir_2
6099 0000C051 C3                  <1>       retn
6100                              <1>
6101                              <1> msftdf_delete_FAT_direntry:
6102 0000C052 8A1D[A5920100]      <1>       mov    bl, [FindFile_LongNameEntryLength]
6103 0000C058 668B0D[D0920100]    <1>       mov    cx, [FindFile_DirEntryNumber]
6104                              <1>       ; ESI = Logical DOS drive description table address
6105                              <1>       ; EDI = Directory buffer entry offset/address
6106 0000C05F E89AFCFFFF          <1>       call   delete_directory_entry
6107 0000C064 721C                <1>       jc     short msftdf_retn
6108                              <1>
6109                              <1> msftdf_dsfde_restore_current_dir_2:
6110 0000C066 803D[833F0100]00    <1>       cmp    byte [Restore_CDIR], 0
6111 0000C06D 7607                <1>       jna    short msftdf_new_dir_fcluster_retn
6112                              <1>       ;mov   esi, [msftdf_drv_offset]
6113 0000C06F E8F1BFFFFF          <1>       call   restore_current_directory
6114 0000C074 720C                <1>       jc     short msftdf_retn
6115                              <1>
6116                              <1> msftdf_new_dir_fcluster_retn:
6117 0000C076 31C9                <1>       xor    ecx, ecx
6118 0000C078 A1[04950100]        <1>       mov    eax, [createfile_FFCluster]
6119 0000C07D BB[1C940100]        <1>       mov    ebx, DestinationFile_Drv
6120                              <1>
6121                              <1> msftdf_retn:
6122 0000C082 C3                  <1>       retn
6123                              <1>
6124                              <1>
6125                              <1> copy_source_file_to_destination_file:
6126                              <1>       ; 17/10/2016
6127                              <1>       ; 16/10/2016
6128                              <1>       ; 15/10/2016
6129                              <1>       ; 30/03/2016, 31/03/2016
6130                              <1>       ; 24/03/2016, 25/03/2016, 28/03/2016
6131                              <1>       ; 21/03/2016, 22/03/2016, 23/03/2016
6132                              <1>       ; 16/03/2016, 17/03/2016, 18/03/2016
6133                              <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
6134                              <1>       ; 02/09/2011 (FILE.ASM 'copy_source_file_to_destination_file')
6135                              <1>       ; 01/08/2010 - 18/05/2011
6136                              <1>       ;
6137                              <1>       ;   Command Interpreter phase 1 enter ->
6138                              <1>       ;           AL = 1 -> Caller is command interpreter
6139                              <1>       ;           AL = 2 -> The second call, re-enter/continue
6140                              <1>       ;   Phase 1 -> Check source file
6141                              <1>       ;              'found' is required
6142                              <1>       ;   Phase 2 -> Check destination file,
6143                              <1>       ;              save 'found' or 'not found' status
6144                              <1>       ;              'permission denied' error will be return
6145                              <1>       ;              if attributes have not for ordinary file
6146                              <1>       ;              without readonly attribute
6147                              <1>       ;   Command Interpreter phase 1 return ->
6148                              <1>       ;              DH = Source file attributes
6149                              <1>       ;              DL = Destination file found status
6150                              <1>       ;              EAX = 0
6151                              <1>       ;   Command Interpreter phase 2 enter ->
6152                              <1>       ;              AL = 2 -> Continue from the last position
6153                              <1>       ;              AH =
6154                              <1>       ;   Phase 3 -> Load source file or use read/write cluster method
6155                              <1>       ;   Phase 4 -> Create destination file if it is not found
6156                              <1>       ;   Phase 5 -> Open destination file
6157                              <1>       ;   Phase 6 -> Read from source and write to destination
6158                              <1>       ;   Phase 7 -> Unload source file, if it is loaded at memory
6159                              <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>
0000C083 3C02           <1>       cmp    al, 2
0000C085 0F845A020000   <1>       je     csftdf2_check_cdrv
                        <1>
                        <1> ; Phase 1
                        <1>
0000C08B A2[C4940100]   <1>       mov    byte [copy_cmd_phase], al
                        <1>
0000C090 57             <1>       push   edi ; *
                        <1>
                        <1> csftdf_parse_sf_path:
0000C091 BF[9C930100]   <1>       mov    edi, SourceFile_Drv
0000C096 E887F4FFFF     <1>       call   parse_path_name
0000C09B 721C           <1>       jc     short csftdf_parse_sf_path_failed
                        <1>
                        <1> csftdf_parse_df_path:
0000C09D 5E             <1>       pop    esi ; * (pushed edi)
                        <1>
                        <1> csftdf_sf_check_filename_exists:
0000C09E 803D[DE930100]21 <1>     cmp    byte [SourceFile_Name], 21h
0000C0A5 7215           <1>       jb     short csftdf_sf_file_not_found_error
                        <1>
0000C0A7 BF[1C940100]   <1>       mov    edi, DestinationFile_Drv
0000C0AC E871F4FFFF     <1>       call   parse_path_name
0000C0B1 7310           <1>       jnc    short csftdf_check_sf_cdrv
                        <1>
0000C0B3 3C01           <1>       cmp    al, 1 ; File or directory name is not existing
0000C0B5 760C           <1>       jna    short csftdf_check_sf_cdrv
                        <1>
                        <1> csftdf_parse_df_path_failed:
0000C0B7 F9             <1>       stc
                        <1> csftdf_sf_error_retn:
0000C0B8 C3             <1>       retn
                        <1>
                        <1> csftdf_parse_sf_path_failed:
0000C0B9 5F             <1>       pop    edi ; *
0000C0BA EBFC           <1>       jmp    short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_sf_file_not_found_error:
0000C0BC B802000000     <1>       mov    eax, 2 ; File not found
0000C0C1 EBF5           <1>       jmp    short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_check_sf_cdrv:
0000C0C3 8A3D[B6890100] <1>       mov    bh, [Current_Drv]
                        <1>
0000C0C9 883D[C7940100] <1>       mov    [csftdf_cdrv], bh ; 23/03/2016
                        <1>
0000C0CF 8A15[9C930100] <1>       mov    dl, [SourceFile_Drv]
0000C0D5 38FA           <1>       cmp    dl, bh ; byte [Current_Drv]
0000C0D7 7407           <1>       je     short csftdf_sf_check_directory
                        <1>
0000C0D9 E8D0BEFFFF     <1>       call   change_current_drive
0000C0DE 72D8           <1>       jc     short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_sf_check_directory:
0000C0E0 BE[9D930100]   <1>       mov    esi, SourceFile_Directory
0000C0E5 803E20         <1>       cmp    byte [esi], 20h
0000C0E8 760F           <1>       jna    short csftdf_find_sf
                        <1>
                        <1> csftdf_sf_change_directory:
0000C0EA FE05[833F0100] <1>       inc    byte [Restore_CDIR]
0000C0F0 30E4           <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000C0F2 E815EEFFFF     <1>       call   change_current_directory
0000C0F7 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:
0000C0F9 BE[DE930100]   <1>       mov    esi, SourceFile_Name
0000C0FE 66B80018       <1>       mov    ax, 1800h ; Except volume label and dirs
0000C102 E84AD2FFFF     <1>       call   find_first_file
0000C107 72AF           <1>       jc     short csftdf_sf_error_retn
                        <1>
                        <1> csftdf_sf_ambgfn_check:
0000C109 6621D2         <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
0000C10C 7407           <1>       jz     short csftdf_sf_found
                        <1>
                        <1> csftdf_ambiguous_file_name_error:
0000C10E B802000000     <1>       mov    eax, 2 ; File not found error
0000C113 F9             <1>       stc
0000C114 C3             <1>       retn
```

```
6265                                <1>
6266                                <1> csftdf_sf_found:
6267 0000C115 A3[C8940100]         <1>         mov     [csftdf_filesize], eax
6268                                <1>
6269 0000C11A 09C0                 <1>         or      eax, eax
6270 0000C11C 7507                 <1>         jnz     short csftdf_set_source_file_direntry
6271                                <1>
6272                                <1> csftdf_sf_file_size_zero:
6273 0000C11E B814000000           <1>         mov     eax, 20 ; TRDOS zero length (file size) error
6274 0000C123 F9                   <1>         stc
6275 0000C124 C3                   <1>         retn
6276                                <1>
6277                                <1> csftdf_set_source_file_direntry:
6278 0000C125 BE[A8920100]         <1>         mov     esi, FindFile_DirEntry
6279 0000C12A BF[EE930100]         <1>         mov     edi, SourceFile_DirEntry
6280 0000C12F B908000000           <1>         mov     ecx, 8
6281 0000C134 F3A5                 <1>         rep     movsd
6282                                <1>
6283                                <1> csftdf_sf_restore_cdrv:
6284                                <1>         ; 22/03/2016
6285 0000C136 8A15[C7940100]       <1>         mov     dl, [csftdf_cdrv]
6286 0000C13C 3A15[B6890100]       <1>         cmp     dl, [Current_Drv]
6287 0000C142 7407                 <1>         je      short csftdf_sf_restore_cdir
6288 0000C144 E865BEFFFF           <1>         call    change_current_drive
6289 0000C149 724F                 <1>         jc      short csftdf_df_error_retn ; 30/03/2016
6290                                <1>
6291                                <1> csftdf_sf_restore_cdir:
6292 0000C14B 803D[833F0100]00     <1>         cmp     byte [Restore_CDIR], 0
6293 0000C152 7612                 <1>         jna     short csftdf_df_check_filename_exists
6294 0000C154 29C0                 <1>         sub     eax, eax
6295 0000C156 BE00010900           <1>         mov     esi, Logical_DOSDisks
6296 0000C15B 88D4                 <1>         mov     ah, dl ; byte [csftdf_cdrv]
6297 0000C15D 01C6                 <1>         add     esi, eax
6298 0000C15F E801BFFFFF           <1>         call    restore_current_directory
6299 0000C164 7234                 <1>         jc      short csftdf_df_error_retn
6300                                <1>
6301                                <1> csftdf_df_check_filename_exists:
6302 0000C166 803D[5E940100]20     <1>         cmp     byte [DestinationFile_Name], 20h
6303 0000C16D 7716                 <1>         ja      short csftdf_check_df_cdrv
6304                                <1>
6305                                <1> csftdf_copy_sf_name:
6306 0000C16F BF[5E940100]         <1>         mov     edi, DestinationFile_Name
6307 0000C174 BE[DE930100]         <1>         mov     esi, SourceFile_Name
6308 0000C179 B10C                 <1>         mov     cl, 12
6309                                <1>
6310                                <1> csftdf_df_copy_sf_name_loop:
6311 0000C17B AC                   <1>         lodsb
6312 0000C17C AA                   <1>         stosb
6313 0000C17D 08C0                 <1>         or      al, al
6314 0000C17F 7404                 <1>         jz      short csftdf_check_df_cdrv
6315 0000C181 FEC9                 <1>         dec     cl
6316 0000C183 75F6                 <1>         jnz     csftdf_df_copy_sf_name_loop
6317                                <1>
6318                                <1> csftdf_check_df_cdrv:
6319 0000C185 8A15[1C940100]       <1>         mov     dl, [DestinationFile_Drv]
6320 0000C18B 3A15[B6890100]       <1>         cmp     dl, [Current_Drv]
6321 0000C191 7408                 <1>         je      short csftdf_df_check_directory
6322                                <1>
6323 0000C193 E816BEFFFF           <1>         call    change_current_drive
6324 0000C198 7301                 <1>         jnc     short csftdf_df_check_directory
6325                                <1>
6326                                <1> csftdf_df_error_retn:
6327 0000C19A C3                   <1>         retn
6328                                <1>
6329                                <1> csftdf_df_check_directory:
6330 0000C19B BE[1D940100]         <1>         mov     esi, DestinationFile_Directory
6331 0000C1A0 803E20               <1>           cmp     byte [esi], 20h
6332 0000C1A3 760F                 <1>         jna     short csftdf_find_df
6333                                <1>
6334                                <1> csftdf_df_change_directory:
6335 0000C1A5 FE05[833F0100]       <1>         inc     byte [Restore_CDIR]
6336 0000C1AB 28E4                 <1>         sub     ah, ah ; CD_COMMAND sign -> 0
6337 0000C1AD E85AEDFFFF           <1>         call    change_current_directory
6338 0000C1B2 72E6                 <1>         jc      short csftdf_df_error_retn
6339                                <1>
6340                                <1> ;csftdf_df_change_prompt_dir_string:
6341                                <1> ;       call    change_prompt_dir_string
6342                                <1>
6343                                <1> csftdf_find_df:
6344                                <1>         ; 23/03/2016
6345 0000C1B4 29DB                 <1>         sub     ebx, ebx
6346 0000C1B6 8A3D[1C940100]       <1>         mov     bh,  [DestinationFile_Drv]
6347 0000C1BC 81C300010900         <1>         add     ebx, Logical_DOSDisks
6348 0000C1C2 891D[F4940100]       <1>         mov     [csftdf_df_drv_dt], ebx
6349                                <1>
6350 0000C1C8 BE[5E940100]         <1>         mov     esi, DestinationFile_Name
6351 0000C1CD 6631C0               <1>         xor     ax, ax
6352                                <1>                 ; DestinationFile_AttributesMask -> any/zero
6353 0000C1D0 E87CD1FFFF           <1>         call    find_first_file
6354 0000C1D5 7218                 <1>         jc      short csftdf_df_check_error_code
6355                                <1>
6356                                <1> csftdf_df_ambgfn_check:
6357 0000C1D7 6609D2               <1>         or      dx, dx ; Ambiguous filename chars used sign (DX>0)
6358 0000C1DA 752A                 <1>         jnz     short csftdf_df_error_inv_fname
6359                                <1>
6360                                <1> csftdf_df_found:
6361 0000C1DC C605[C6940100]01     <1>         mov     byte [DestinationFileFound], 1
6362                                <1>         ; 17/10/2016 (cl -> bl)
6363 0000C1E3 80E31F               <1>         and     bl, 1Fh ; Attributes, D-V-S-H-R
6364 0000C1E6 745F                 <1>         jz      short csftdf_df_save_first_cluster
6365                                <1>
6366                                <1> csftdf_df_permission_denied_retn:
6367 0000C1E8 B805000000           <1>         mov     eax, 05h ; Access/Permission denied.
6368                                <1> csftdf_df_error_stc_retn:
6369 0000C1ED F9                   <1>         stc
```

```
6370 0000C1EE C3                <1>       retn
6371                            <1>
6372                            <1> csftdf_df_check_error_code:
6373                            <1>       ;cmp   eax, 2
6374 0000C1EF 3C02              <1>       cmp    al, 2
6375 0000C1F1 75FA              <1>       jne    short csftdf_df_error_stc_retn
6376                            <1>
6377 0000C1F3 C605[C6940100]00  <1>       mov    byte [DestinationFileFound], 0
6378                            <1>
6379                            <1>       ; 15/10/2016
6380 0000C1FA BE[98920100]      <1>       mov    esi, FindFile_Name ; *
6381 0000C1FF E80BD5FFFF        <1>       call   check_filename
6382 0000C204 7307              <1>       jnc    short csftdf_df_valid_fname
6383                            <1> csftdf_df_error_inv_fname: ; 'invalid file name !'
6384 0000C206 B81A000000        <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
6385 0000C20B F9                <1>       stc
6386 0000C20C C3                <1>       retn
6387                            <1>
6388                            <1> csftdf_df_valid_fname:
6389                            <1>       ; 21/03/2016
6390                            <1>       ; (Capitalized file name)
6391                            <1>       ;mov   esi, FindFile_Name ; * ; 15/10/2016
6392 0000C20D BF[5E940100]      <1>       mov    edi, DestinationFile_Name
6393 0000C212 A5                <1>       movsd
6394 0000C213 A5                <1>       movsd
6395 0000C214 A5                <1>       movsd
6396                            <1>       ;movsb
6397                            <1>
6398                            <1> csftdf_check_disk_free_size_0:
6399 0000C215 A1[0A940100]      <1>       mov    eax, [SourceFile_DirEntry+DirEntry_FileSize]
6400                            <1>
6401                            <1> csftdf_check_disk_free_size_1:
6402                            <1>       ;sub   ebx, ebx
6403                            <1>       ;mov   esi, Logical_DOSDisks
6404                            <1>       ;mov   bh,  [DestinationFile_Drv]
6405                            <1>       ;add   esi, ebx
6406                            <1>
6407 0000C21A 8B35[F4940100]    <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
6408                            <1>
6409 0000C220 0FB74E11          <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
6410 0000C224 01C8              <1>       add    eax, ecx
6411 0000C226 48                <1>       dec    eax  ; file size (additional bytes) + 511 (round up)
6412                            <1> csftdf_check_disk_free_size_3: ; 16/03/2016
6413 0000C227 29D2              <1>       sub    edx, edx
6414 0000C229 F7F1              <1>       div    ecx ; bytes per sector
6415                            <1>
6416                            <1> csftdf_check_disk_free_size:
6417 0000C22B 3B4674            <1>       cmp    eax, [esi+LD_FreeSectors]
6418 0000C22E 0F8294000000      <1>         jb     csftdf_check_disk_free_size_ok
6419 0000C234 770A              <1>       ja     short csftdf_df_insufficient_disk_space
6420                            <1>
6421 0000C236 807E0300          <1>       cmp    byte [esi+LD_FATType], 0 ; FS needs FDT sector also.
6422 0000C23A 0F8788000000      <1>         ja     csftdf_check_disk_free_size_ok
6423                            <1>
6424                            <1> csftdf_df_insufficient_disk_space:
6425 0000C240 B827000000        <1>       mov    eax, 27h ; insufficient disk space
6426 0000C245 EBA6              <1>       jmp    short csftdf_df_error_stc_retn
6427                            <1>
6428                            <1> csftdf_df_save_first_cluster:
6429                            <1>       ; ESI = FindFile_DirEntry (for the old destination file)
6430                            <1>       ; EAX = Old destination file size
6431                            <1>       ; 24/03/2016
6432                            <1>       ; EDI = Directory entry address (within Dir Buffer boundaries)
6433 0000C247 81EF00000800      <1>       sub    edi, Directory_Buffer  ; (<65536)
6434 0000C24D 66C1EF05          <1>       shr    di, 5 ; Convert entry offset to entry index/number
6435 0000C251 66893D[96940100]  <1>       mov    [DestinationFile_DirEntryNumber], di ; (<2048)
6436                            <1>
6437                            <1> csftdf_df_check_sf_df_fcluster:
6438 0000C258 668B5614          <1>       mov    dx, [esi+DirEntry_FstClusHI]
6439 0000C25C C1E210            <1>       shl    edx, 16
6440 0000C25F 668B561A          <1>       mov    dx, [esi+DirEntry_FstClusLO]
6441 0000C263 8915[D8940100]    <1>       mov    [csftdf_df_cluster], edx
6442                            <1> csftdf_df_check_sf_df_fcluster_1:
6443 0000C269 668B15[02940100]  <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
6444 0000C270 C1E210            <1>       shl    edx, 16
6445 0000C273 668B15[08940100]  <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
6446 0000C27A 3B15[D8940100]    <1>       cmp    edx, [csftdf_df_cluster]
6447 0000C280 7512              <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
6448                            <1> csftdf_df_check_sf_df_drv:
6449 0000C282 8A15[9C930100]    <1>       mov    dl, [SourceFile_Drv]
6450 0000C288 3A15[1C940100]    <1>       cmp    dl, [DestinationFile_Drv]
6451 0000C28E 7504              <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
6452                            <1>
6453                            <1>       ; source and destination files are same !
6454                            <1>       ; (they have same first cluster value on same logical disk)
6455                            <1>
6456 0000C290 31C0              <1>       xor    eax, eax ; mov eax, 0 -> Bad command or file name !
6457 0000C292 F9                <1>       stc
6458 0000C293 C3                <1>       retn
6459                            <1>
6460                            <1> csftdf_df_check_sf_df_fcluster_ok:
6461                            <1> csftdf_df_move_findfile_struct:
6462                            <1>       ; mov  esi, FindFile_DirEntry
6463 0000C294 BF[6E940100]      <1>       mov    edi, DestinationFile_DirEntry
6464 0000C299 B908000000        <1>       mov    ecx, 8
6465 0000C29E F3A5              <1>       rep    movsd
6466                            <1>
6467                            <1> csftdf_check_disk_free_size_2:
6468 0000C2A0 89C2              <1>       mov    edx, eax ; Old destination file size
6469                            <1>
6470                            <1>       ;mov   eax, [SourceFile_DirEntry+DirEntry_FileSize]
6471 0000C2A2 A1[C8940100]      <1>       mov    eax, [csftdf_filesize] ; 23/03/2016
6472                            <1>
6473                            <1>       ;;sub  ecx, ecx ; 0
6474                            <1>       ;mov   esi, Logical_DOSDisks
```

```
                                <1>       ;mov   ch,  [DestinationFile_Drv]
                                <1>       ;add   esi, ecx
                                <1>       ;
                                <1>       ;mov   [csftdf_df_drv_dt], esi
                                <1>
0000C2A7 8B35[F4940100]         <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
                                <1>
0000C2AD 668B4E11               <1>       mov    cx, [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
0000C2B1 01CA                   <1>       add    edx, ecx ; + 512
0000C2B3 01C8                   <1>       add    eax, ecx ; + 512
0000C2B5 4A                     <1>       dec    edx ; old file size + 511 (round up)
0000C2B6 48                     <1>       dec    eax ; new file size + 511 (round up)
0000C2B7 F7D9                   <1>       neg    ecx ; -512 ; 0FFFFFE00h
0000C2B9 21CA                   <1>       and    edx, ecx ; = old sector count * 512
0000C2BB 21C8                   <1>       and    eax, ecx ; = new sector count * 512
                                <1>
0000C2BD 29D0                   <1>       sub    eax, edx ; new file size - old file size (on disk)
0000C2BF 7607                   <1>       jna    short csftdf_check_disk_free_size_ok
                                <1>
0000C2C1 F7D9                   <1>       neg    ecx ; 512 (bytes per sector) ; 200h
                                <1>       ; check free space for additional sectors
                                <1>       ; eax = number of additional sectors * bytes per sector
                                <1>       ; esi = Logical DOS drive number (of destination disk)
0000C2C3 E95FFFFFFF             <1>         jmp     csftdf_check_disk_free_size_3
                                <1>
                                <1> csftdf_check_disk_free_size_ok:
                                <1>       ; 18/03/2016
                                <1> csftdf_df_check_copy_cmd_phase:
0000C2C8 A0[C4940100]           <1>       mov    al, [copy_cmd_phase]
0000C2CD 3C01                   <1>       cmp    al, 1
0000C2CF 7514                   <1>       jne    short csftdf2_check_cdrv
                                <1>
0000C2D1 31C0                   <1>       xor    eax, eax
0000C2D3 A2[C4940100]           <1>       mov    [copy_cmd_phase], al ; 0
                                <1>
0000C2D8 8A15[C6940100]         <1>       mov    dl, [DestinationFileFound]
0000C2DE 8A35[F9930100]         <1>       mov    dh, [SourceFile_DirEntry+11] ; Attributes
                                <1>
                                <1> csftdf_return:
0000C2E4 C3                     <1>       retn
                                <1>
                                <1> ; Phase 2
                                <1>
                                <1> csftdf2_check_cdrv:
                                <1>       ; 18/03/2016
                                <1>       ; Here, destination drive and directory are ready !
                                <1>       ; (checking/restoring is not needed)
                                <1>       ; (Since at the end of the phase 1)
                                <1>
                                <1> ;     mov    dl, [DestinationFile_Drv]
                                <1> ;     cmp    dl, [Current_Drv]
                                <1> ;     je     short csftdf2_df_check_directory
                                <1> ;
                                <1> ;     call   change_current_drive
                                <1> ;     jc     short csftdf2_read_error
                                <1> ;
                                <1> ;csftdf2_df_check_directory:
                                <1> ;     mov    esi, DestinationFile_Directory
                                <1> ;     cmp    byte [esi], 20h
                                <1> ;     jna    short csftdf2_df_check_found_or_not
                                <1> ;
                                <1> ;csftdf2_df_change_directory:
                                <1> ;     inc    byte [Restore_CDIR]
                                <1> ;     xor    ah, ah ; CD_COMMAND sign -> 0
                                <1> ;     call   change_current_directory
                                <1> ;     jc     short csftdf2_stc_return
                                <1> ;
                                <1> ;;csftdf2_df_change_prompt_dir_string:
                                <1> ;;    call   change_prompt_dir_string
                                <1>
                                <1> csftdf2_df_check_found_or_not:
                                <1>       ; 21/03/2016
0000C2E5 803D[C6940100]00       <1>       cmp    byte [DestinationFileFound], 0
0000C2EC 7739                   <1>       ja     short csftdf2_set_sf_percentage
                                <1>
                                <1> csftdf2_create_file:
0000C2EE BE[5E940100]           <1>       mov    esi, DestinationFile_Name
0000C2F3 A1[C8940100]           <1>       mov    eax, [csftdf_filesize]
0000C2F8 30C9                   <1>       xor    cl, cl ; 0
                                <1>
0000C2FA 31DB                   <1>       xor    ebx, ebx ; 0
0000C2FC 4B                     <1>       dec    ebx ; 0FFFFFFFFh
                                <1>
                                <1>       ; INPUT ->
                                <1>       ;      EAX -> File Size
                                <1>       ;      ESI = ASCIIZ File name
                                <1>       ;       CL = File attributes
                                <1>       ;      EBX = FFFFFFFFh -> empty file sign for FAT fs
                                <1>       ;      EBX <> FFFFFFFFh -> use file size for FAT fs
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = New file's first cluster
                                <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
                                <1>       ;      EBX = CreateFile_Size address
                                <1>       ;      ECX = Sectors per cluster (<256)
                                <1>       ;      EDX = Directory Entry Index/Number (<65536)
                                <1>       ;
                                <1>       ;      cf = 1 -> error code in AL (EAX)
                                <1>
0000C2FD E8EC050000             <1>       call   create_file
                                <1>       ;pop   esi
0000C302 0F82A3050000           <1>         jc      csftdf2_rw_error
                                <1>
                                <1> csftdf2_create_file_OK:
0000C308 A3[D8940100]           <1>       mov    [csftdf_df_cluster], eax
```

```
                                <1>
                                <1>       ; 24/03/2016
0000C30D 668915[96940100]       <1>       mov   [DestinationFile_DirEntryNumber], dx
                                <1>
                                <1>       ; 21/03/2016
0000C314 BE00000800             <1>       mov   esi, Directory_Buffer
0000C319 C1E205                 <1>       shl   edx, 5 ; 32 * index number
0000C31C 01D6                   <1>       add   esi, edx
0000C31E BF[6E940100]           <1>       mov   edi, DestinationFile_DirEntry
0000C323 B108                   <1>       mov   cl, 8 ; 32 bytes
0000C325 F3A5                   <1>       rep   movsd
                                <1>
                                <1> csftdf2_set_sf_percentage:
                                <1>       ; 17/03/2016
0000C327 31C0                   <1>       xor   eax, eax
0000C329 A2[EC940100]           <1>       mov   [csftdf_percentage], al ; 0, reset
                                <1>
0000C32E A3[E4940100]           <1>       mov   [csftdf_sf_rbytes], eax ; 0, reset
0000C333 A3[E8940100]           <1>       mov   [csftdf_df_wbytes], eax ; 0, reset
                                <1>
0000C338 8A25[9C930100]         <1>       mov   ah, [SourceFile_Drv]
0000C33E BE00010900             <1>       mov   esi, Logical_DOSDisks
0000C343 01C6                   <1>       add   esi, eax
                                <1>
0000C345 8935[F0940100]         <1>       mov   [csftdf_sf_drv_dt], esi ; 23/03/2016
                                <1>
0000C34B 668B15[02940100]       <1>       mov   dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
0000C352 C1E210                 <1>       shl   edx, 16
0000C355 668B15[08940100]       <1>       mov   dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
0000C35C 8915[D4940100]         <1>       mov   [csftdf_sf_cluster], edx
                                <1>
                                <1>       ; 16/03/2016
                                <1>       ; Note: Singlix FS boot sector parameters (for cluster
                                <1>       ;     related calculations) has same offset
                                <1>       ;     values from LD_BPB as in FAT file system.
                                <1>       ;     [esi+LD_BPB+SecPerClust] is 1 for Singlix FS.
                                <1>       ;
0000C362 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
0000C366 880D[1A940100]         <1>       mov   [SourceFile_SecPerClust], cl
                                <1>
                                <1>       ; 17/03/2016
0000C36C 386E03                 <1>       cmp   [esi+LD_FATType], ch ; 0
0000C36F 7707                   <1>       ja    short csftdf2_set_sf_percent_rsize1
                                <1>
0000C371 B800000100             <1>       mov   eax, 65536 ; read/write buffer size for Singlix FS
0000C376 EB06                   <1>       jmp   short csftdf2_set_sf_percent_rsize2
                                <1>
                                <1> csftdf2_set_sf_percent_rsize1:
0000C378 668B4611               <1>       mov   ax, [esi+LD_BPB+BytesPerSec]
0000C37C F7E1                   <1>       mul   ecx
                                <1>       ;sub  edx, edx
                                <1> csftdf2_set_sf_percent_rsize2:
0000C37E A3[DC940100]           <1>       mov   [csftdf_r_size], eax
                                <1>
                                <1> csftdf2_set_df_percentage:
                                <1>       ;sub  eax, eax
                                <1>       ;mov  ah, [DestinationFile_Drv]
                                <1>       ;mov  edi, Logical_DOSDisks
                                <1>       ;add  edi, eax
                                <1>       ;mov  [csftdf_df_drv_dt], edi ; 17/03/2016
                                <1>
0000C383 8B3D[F4940100]         <1>       mov   edi, [csftdf_df_drv_dt] ; 23/03/2016
                                <1>
                                <1>       ; 16/03/2016
                                <1>       ; Note: Singlix FS boot sector parameters (for cluster
                                <1>       ;     related calculations) has same offset
                                <1>       ;     values from LD_BPB as in FAT file system.
                                <1>       ;     [edi+LD_BPB+SecPerClust] is 1 for Singlix FS.
                                <1>       ;
                                <1>       ;movzx ecx, byte [edi+LD_BPB+SecPerClust]
0000C389 8A4F13                 <1>       mov   cl, [edi+LD_BPB+SecPerClust]
0000C38C 880D[9A940100]         <1>       mov   [DestinationFile_SecPerClust], cl
                                <1>
                                <1>       ; 17/03/2016
0000C392 386F03                 <1>       cmp   [edi+LD_FATType], ch ; 0
0000C395 7707                   <1>       ja    short csftdf2_set_df_percent_wsize1
                                <1>
0000C397 B800000100             <1>       mov   eax, 65536 ; read/write buffer size for Singlix FS
0000C39C EB06                   <1>       jmp   short csftdf2_set_df_percent_wsize2
                                <1>
                                <1> csftdf2_set_df_percent_wsize1:
0000C39E 0FB74711               <1>       movzx  eax, word [edi+LD_BPB+BytesPerSec]
0000C3A2 F7E1                   <1>       mul   ecx
                                <1>       ;sub  edx, edx
                                <1> csftdf2_set_df_percent_wsize2:
0000C3A4 A3[E0940100]           <1>       mov   [csftdf_w_size], eax
                                <1>
0000C3A9 A1[C8940100]           <1>       mov   eax, [csftdf_filesize]
                                <1>
0000C3AE 3D00000100             <1>       cmp   eax, 65536 ; 64KB   ; small file
0000C3B3 721F                   <1>       jb    short csftdf2_load_file ; do not display percentage
                                <1>
                                <1> csftdf2_reset_wf_percent_ptr_chk_64k:
0000C3B5 B201                   <1>       mov   dl, 1 ; 25/03/2016
                                <1>
0000C3B7 3D00000400             <1>       cmp   eax, 65536*4 ; 256KB
0000C3BC 7310                   <1>       jnb   short csftdf2_enable_percentage_display ; big file
                                <1>
                                <1>       ; 64-128KB file size for floppy disks
0000C3BE 3815[9C930100]         <1>       cmp   byte [SourceFile_Drv], dl ; 1 ; read from floppy disk ?
0000C3C4 7608                   <1>       jna   short csftdf2_enable_percentage_display
                                <1>
0000C3C6 3815[1C940100]         <1>       cmp   byte [DestinationFile_Drv], dl ; 1 ; write to floppy disk ?
0000C3CC 7706                   <1>       ja    short csftdf2_load_file
                                <1>
```

```
                                <1> csftdf2_enable_percentage_display:
0000C3CE 8815[EC940100]         <1>       mov   [csftdf_percentage], dl ; 1
                                <1>
                                <1> csftdf2_load_file:
                                <1>       ; 13/05/2016
                                <1>       ; 19/03/2016
                                <1>       ; 18/03/2016
                                <1>       ; 17/03/2016
0000C3D4 B40F                   <1>       mov   ah, 0Fh
0000C3D6 E82853FFFF             <1>       call  _int10h
                                <1>       ; 13/05/2016
0000C3DB 883D[ED940100]         <1>       mov   [csftdf_videopage], bh ; active video page
0000C3E1 B403                   <1>       mov   ah, 03h
0000C3E3 E81B53FFFF             <1>       call  _int10h
0000C3E8 668915[EE940100]       <1>       mov   [csftdf_cursorpos], dx
                                <1>
0000C3EF 29C0                   <1>       sub   eax, eax
0000C3F1 A2[C5940100]           <1>       mov   [csftdf_rw_err], al ; 0
                                <1>
                                <1> ; ///
                                <1> csftdf_sf_amb: ; 15/03/2016
0000C3F6 8B0D[C8940100]         <1>       mov   ecx, [csftdf_filesize]    ; 23/03/2016
                                <1>
                                <1>       ; TRDOS 386 (TRDOS v2.0)
                                <1>       ; Allocate contiguous memory block for loading the file
                                <1>
                                <1>       ;mov  ecx, [SourceFile_DirEntry+DirEntry_FileSize]
                                <1>
                                <1>       ;sub  eax, eax ; First free memory aperture
                                <1>
                                <1>       ; eax = 0 (Allocate memory from the beginning)
                                <1>       ; ecx = File (Allocation) size in bytes
                                <1>
0000C3FC E8509FFFFF             <1>       call  allocate_memory_block
0000C401 7304                   <1>       jnc   short loc_check_sf_save_loading_parms
                                <1>
0000C403 29C0                   <1>       sub   eax, eax
0000C405 29C9                   <1>       sub   ecx, ecx
                                <1>
                                <1> loc_check_sf_save_loading_parms:
0000C407 A3[CC940100]           <1>       mov   [csftdf_sf_mem_addr], eax ; loading address
0000C40C 890D[D0940100]         <1>       mov   [csftdf_sf_mem_bsize], ecx ; block size
                                <1> ; ///
                                <1>       ; 19/03/2016
0000C412 8B35[F0940100]         <1>       mov   esi, [csftdf_sf_drv_dt] ; logical dos drv desc. tbl.
                                <1>
                                <1>       ; 17/03/2016
0000C418 09C0                   <1>       or    eax, eax ; contiguous free memory block address
0000C41A 0F845B010000           <1>         jz      csftdf2_read_sf_cluster
                                <1>
                                <1>       ; 18/03/2016
0000C420 8B1D[CC940100]         <1>       mov   ebx, [csftdf_sf_mem_addr] ; memory block address
                                <1>
0000C426 807E0300               <1>       cmp   byte [esi+LD_FATType], 0
0000C42A 0F8605020000           <1>         jna     csftdf2_load_fs_file
                                <1>
                                <1> csftdf2_load_fat_file:
0000C430 53                     <1>       push  ebx ; *
                                <1>
                                <1> csftdf2_load_fat_file_next:
0000C431 BE[D3450100]           <1>       mov   esi, msg_reading
0000C436 E8EAAFFFFF             <1>       call  print_msg
                                <1>
0000C43B 803D[EC940100]00       <1>       cmp   byte [csftdf_percentage], 0
0000C442 7605                   <1>       jna   short csftdf2_load_fat_file_1
                                <1>
0000C444 E87C000000             <1>       call  csftdf2_print_percentage ; 19/03/2016
                                <1>
                                <1> csftdf2_load_fat_file_1:
0000C449 8B35[F0940100]         <1>       mov   esi, [csftdf_sf_drv_dt]
0000C44F 5B                     <1>       pop   ebx ; *
                                <1>
                                <1> csftdf2_load_fat_file_2:
0000C450 E8B8000000             <1>       call  csftdf2_read_fat_file_sectors ; 19/03/2016
0000C455 0F8250040000           <1>         jc      csftdf2_rw_error ; eocc! or disk error!
                                <1>
0000C45B 09D2                   <1>       or    edx, edx ; edx > 0 -> EOF
0000C45D 7520                   <1>       jnz   short csftdf2_load_fat_file_ok
                                <1>
0000C45F 803D[EC940100]00       <1>       cmp   byte [csftdf_percentage], 0
0000C466 76E8                   <1>       jna   short csftdf2_load_fat_file_2
                                <1>
0000C468 53                     <1>       push  ebx ; *
                                <1>
                                <1>       ; Set cursor position
                                <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
0000C469 8A3D[ED940100]         <1>       mov   bh, [csftdf_videopage]
0000C46F 668B15[EE940100]       <1>       mov   dx, [csftdf_cursorpos]
0000C476 B402                   <1>       mov   ah, 2
0000C478 E88652FFFF             <1>       call  _int10h
0000C47D EBB2                   <1>       jmp   short csftdf2_load_fat_file_next
                                <1>
                                <1> csftdf2_load_fat_file_ok:
0000C47F 803D[EC940100]00       <1>       cmp   byte [csftdf_percentage], 0
0000C486 0F8651020000           <1>         jna     csftdf2_save_file ; 25/03/2016
                                <1>
                                <1>       ; "Reading... 100%"
0000C48C BF[EB450100]           <1>       mov   edi, percentagestr
0000C491 B031                   <1>       mov   al, '1'
0000C493 AA                     <1>       stosb
0000C494 B030                   <1>       mov   al, '0'
0000C496 AA                     <1>       stosb
0000C497 AA                     <1>       stosb
                                <1>
0000C498 8A3D[ED940100]         <1>       mov   bh, [csftdf_videopage]
```

```
6790 0000C49E 668B15[EE940100]    <1>       mov    dx, [csftdf_cursorpos]
6791 0000C4A5 B402                <1>       mov    ah, 2
6792 0000C4A7 E85752FFFF          <1>       call   _int10h
6793                              <1>
6794 0000C4AC BE[D3450100]        <1>       mov    esi, msg_reading
6795 0000C4B1 E86FAFFFFF          <1>       call   print_msg
6796                              <1>
6797 0000C4B6 BE[EB450100]        <1>       mov    esi, percentagestr
6798 0000C4BB E865AFFFFF          <1>       call   print_msg
6799                              <1>
6800 0000C4C0 E918020000          <1>         jmp     csftdf2_save_file ; 25/03/2016
6801                              <1>
6802                              <1> csftdf2_print_percentage:
6803                              <1>       ; 09/12/2017
6804                              <1>       ; 19/03/2016
6805                              <1>       ; 18/03/2016
6806 0000C4C5 B020                <1>       mov    al, 20h
6807 0000C4C7 BF[EB450100]        <1>       mov    edi, percentagestr
6808 0000C4CC AA                  <1>       stosb
6809 0000C4CD AA                  <1>       stosb
6810 0000C4CE A1[E4940100]        <1>       mov    eax, [csftdf_sf_rbytes]
6811 0000C4D3 BA64000000          <1>       mov    edx, 100
6812 0000C4D8 F7E2                <1>       mul    edx
6813 0000C4DA 8B0D[C8940100]      <1>       mov    ecx, [csftdf_filesize]
6814 0000C4E0 F7F1                <1>       div    ecx
6815 0000C4E2 B10A                <1>       mov    cl, 10
6816 0000C4E4 F6F1                <1>       div    cl
6817 0000C4E6 80C430              <1>       add    ah, '0'
6818 0000C4E9 8827                <1>       mov    [edi], ah
6819 0000C4EB 20C0                <1>       and    al, al
6820 0000C4ED 740A                <1>       jz     short csftdf2_print_percent_1
6821 0000C4EF 4F                  <1>       dec    edi
6822                              <1>       ;cbw
6823 0000C4F0 28E4                <1>       sub    ah, ah ; 09/12/2017
6824 0000C4F2 F6F1                <1>       div    cl
6825 0000C4F4 80C430              <1>       add    ah, '0'
6826 0000C4F7 8827                <1>       mov    [edi], ah
6827                              <1>       ;and   al, al
6828                              <1>       ;jz    short csftdf2_print_percent_1
6829                              <1>       ;dec   edi
6830                              <1>       ;mov   [edi], '1' ; 100%
6831                              <1>
6832                              <1> csftdf2_print_percent_1:
6833 0000C4F9 BE[EB450100]        <1>       mov    esi, percentagestr
6834                              <1>       ;call  print_msg
6835                              <1>       ;retn
6836 0000C4FE E922AFFFFF          <1>       jmp    print_msg
6837                              <1>
6838                              <1> csftdf2_read_file_sectors:
6839                              <1>       ; 19/03/2016
6840 0000C503 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
6841 0000C507 0F8627070000        <1>         jna     csftdf2_read_fs_file_sectors
6842                              <1>
6843                              <1> csftdf2_read_fat_file_sectors:
6844                              <1>       ; 19/03/2016
6845                              <1>       ; 18/03/2016
6846                              <1>       ; return:
6847                              <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
6848                              <1>       ;   CF = 0 & EDX = 0 -> not EOF
6849                              <1>       ;   CF = 1 -> read error (error code in AL)
6850                              <1>
6851                              <1> csftdf2_read_fat_file_secs_0:
6852 0000C50D 8B15[C8940100]      <1>       mov    edx, [csftdf_filesize]
6853 0000C513 2B15[E4940100]      <1>       sub    edx, [csftdf_sf_rbytes]
6854 0000C519 3B15[DC940100]      <1>       cmp    edx, [csftdf_r_size]
6855 0000C51F 7306                <1>       jnb    short csftdf2_read_fat_file_secs_1
6856 0000C521 8915[DC940100]      <1>       mov    [csftdf_r_size], edx
6857                              <1>
6858                              <1> csftdf2_read_fat_file_secs_1:
6859 0000C527 A1[DC940100]        <1>       mov    eax, [csftdf_r_size]
6860 0000C52C 29D2                <1>       sub    edx, edx
6861 0000C52E 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
6862 0000C532 01C8                <1>       add    eax, ecx
6863 0000C534 48                  <1>       dec    eax
6864 0000C535 F7F1                <1>       div    ecx
6865 0000C537 89C1                <1>       mov    ecx, eax ; sector count
6866 0000C539 A1[D4940100]        <1>       mov    eax, [csftdf_sf_cluster]
6867                              <1>
6868                              <1>       ; EBX = memory block address (current)
6869                              <1>
6870 0000C53E E821090000          <1>       call   read_fat_file_sectors
6871 0000C543 7235                <1>       jc     short csftdf2_read_fat_file_secs_3
6872                              <1>
6873                              <1>       ; EBX = next memory address
6874                              <1>
6875 0000C545 A1[E4940100]        <1>       mov    eax, [csftdf_sf_rbytes]
6876 0000C54A 0305[DC940100]      <1>       add    eax, [csftdf_r_size]
6877 0000C550 8B15[C8940100]      <1>       mov    edx, [csftdf_filesize]
6878 0000C556 39D0                <1>       cmp    eax, edx
6879 0000C558 7320                <1>       jnb    short csftdf2_read_fat_file_secs_3 ; edx > 0
6880 0000C55A A3[E4940100]        <1>       mov    [csftdf_sf_rbytes], eax
6881                              <1>
6882 0000C55F 53                  <1>       push   ebx ; *
6883                              <1>       ; get next cluster (csftdf_r_size! bytes)
6884 0000C560 A1[D4940100]        <1>       mov    eax, [csftdf_sf_cluster]
6885 0000C565 E8CC060000          <1>       call   get_next_cluster
6886 0000C56A 5B                  <1>       pop    ebx ; *
6887 0000C56B 7306                <1>       jnc    short csftdf2_read_fat_file_secs_2
6888                              <1>
6889                              <1>       ; 15/10/2016
6890                              <1>       ;Disk read error instad of drv not ready err
6891 0000C56D B811000000          <1>       mov    eax, 17 ; Read error !
6892 0000C572 C3                  <1>       retn
6893                              <1>
6894                              <1> csftdf2_read_fat_file_secs_2:
```

```
6895 0000C573 29D2              <1>       sub    edx, edx ; 0
6896 0000C575 A3[D4940100]      <1>       mov    [csftdf_sf_cluster], eax ; next cluster
6897                            <1>
6898                            <1> csftdf2_read_fat_file_secs_3:
6899 0000C57A C3                <1>       retn
6900                            <1>
6901                            <1> csftdf2_read_sf_cluster:
6902                            <1>       ; 19/03/2016
6903 0000C57B BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
6904                            <1>
6905 0000C580 803D[EC940100]00  <1>       cmp    byte [csftdf_percentage], 0
6906 0000C587 760D              <1>       jna    short csftdf2_read_sf_clust_2
6907                            <1>
6908 0000C589 53                <1>       push   ebx ; *
6909                            <1>
6910                            <1> csftdf2_read_sf_clust_next:
6911 0000C58A E836FFFFFF        <1>       call   csftdf2_print_percentage
6912                            <1>
6913                            <1> csftdf2_read_sf_clust_0:
6914 0000C58F 8B35[F0940100]    <1>       mov    esi, [csftdf_sf_drv_dt]
6915                            <1> csftdf2_read_sf_clust_1:
6916 0000C595 5B                <1>       pop    ebx ; *
6917                            <1>
6918                            <1> csftdf2_read_sf_clust_2:
6919 0000C596 89DA              <1>       mov    edx, ebx
6920 0000C598 0315[DC940100]    <1>       add    edx, [csftdf_r_size]
6921 0000C59E 81FA00000800      <1>       cmp    edx, Cluster_Buffer + 65536
6922 0000C5A4 772F              <1>       ja     short csftdf2_write_df_cluster
6923                            <1>
6924 0000C5A6 E858FFFFFF        <1>       call   csftdf2_read_file_sectors ; 19/03/2016
6925 0000C5AB 0F8280020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
6926                            <1>
6927 0000C5B1 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
6928 0000C5B3 7520              <1>       jnz    short csftdf2_write_df_cluster
6929                            <1>
6930 0000C5B5 803D[EC940100]00  <1>       cmp    byte [csftdf_percentage], 0
6931 0000C5BC 76D8              <1>       jna    short csftdf2_read_sf_clust_2
6932                            <1>
6933 0000C5BE 53                <1>       push   ebx ; *
6934                            <1>
6935                            <1>       ; Set cursor position
6936                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
6937 0000C5BF 8A3D[ED940100]    <1>       mov    bh, [csftdf_videopage]
6938 0000C5C5 668B15[EE940100]  <1>       mov    dx, [csftdf_cursorpos]
6939 0000C5CC B402              <1>       mov    ah, 2
6940 0000C5CE E83051FFFF        <1>       call   _int10h
6941 0000C5D3 EBB5              <1>       jmp    short csftdf2_read_sf_clust_next
6942                            <1>
6943                            <1> csftdf2_write_df_cluster:
6944                            <1>       ; 19/03/2016
6945 0000C5D5 8B35[F4940100]    <1>       mov    esi, [csftdf_df_drv_dt]
6946 0000C5DB BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
6947                            <1>
6948                            <1> csftdf2_write_df_clust_next:
6949 0000C5E0 E855000000        <1>       call   csftdf2_write_file_sectors ; 19/03/2016
6950 0000C5E5 0F8246020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
6951                            <1>
6952 0000C5EB 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
6953 0000C5ED 750A              <1>       jnz    short csftdf2_rw_f_clust_ok
6954                            <1>
6955 0000C5EF 81FB00000800      <1>       cmp    ebx, Cluster_Buffer + 65536
6956 0000C5F5 72E9              <1>       jb     short csftdf2_write_df_clust_next
6957                            <1>
6958 0000C5F7 EB82              <1>       jmp    short csftdf2_read_sf_cluster
6959                            <1>
6960                            <1> csftdf2_rw_f_clust_ok:
6961 0000C5F9 803D[EC940100]00  <1>       cmp    byte [csftdf_percentage], 0
6962 0000C600 0F86B2010000      <1>         jna     csftdf2_save_fat_file_4 ; 25/03/2016
6963                            <1>
6964                            <1>       ; "100%"
6965 0000C606 BF[EB450100]      <1>       mov    edi, percentagestr
6966 0000C60B B031              <1>       mov    al, '1'
6967 0000C60D AA                <1>       stosb
6968 0000C60E B030              <1>       mov    al, '0'
6969 0000C610 AA                <1>       stosb
6970 0000C611 AA                <1>       stosb
6971                            <1>
6972 0000C612 8A3D[ED940100]    <1>       mov    bh, [csftdf_videopage]
6973 0000C618 668B15[EE940100]  <1>       mov    dx, [csftdf_cursorpos]
6974 0000C61F B402              <1>       mov    ah, 2
6975 0000C621 E8DD50FFFF        <1>       call   _int10h
6976                            <1>
6977 0000C626 BE[EB450100]      <1>       mov    esi, percentagestr
6978 0000C62B E8F5ADFFFF        <1>       call   print_msg
6979                            <1>
6980 0000C630 E983010000        <1>         jmp     csftdf2_save_fat_file_4
6981                            <1>
6982                            <1> csftdf2_load_fs_file:
6983                            <1>       ; temporary - 18/03/2016
6984 0000C635 E96F020000        <1>         jmp     csftdf2_read_error
6985                            <1>
6986                            <1> csftdf2_write_file_sectors:
6987                            <1>       ; 19/03/2016
6988 0000C63A 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
6989 0000C63E 0F86F1050000      <1>         jna     csftdf2_write_fs_file_sectors
6990                            <1>
6991                            <1> csftdf2_write_fat_file_sectors:
6992                            <1>       ; 19/03/2016
6993                            <1>       ; 18/03/2016
6994                            <1>       ; return:
6995                            <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
6996                            <1>       ;   CF = 0 & EDX = 0 -> not EOF
6997                            <1>       ;   CF = 1 -> write error (error code in AL)
6998                            <1>
6999                            <1> csftdf2_write_fat_file_secs_0:
```

```
7000 0000C644 8B15[C8940100]      <1>       mov    edx, [csftdf_filesize]
7001 0000C64A 2B15[E8940100]      <1>       sub    edx, [csftdf_df_wbytes]
7002 0000C650 3B15[E0940100]      <1>       cmp    edx, [csftdf_w_size]
7003 0000C656 7306                <1>       jnb    short csftdf2_write_fat_file_secs_1
7004 0000C658 8915[E0940100]      <1>       mov    [csftdf_w_size], edx
7005                              <1>
7006                              <1> csftdf2_write_fat_file_secs_1:
7007 0000C65E A1[E0940100]        <1>       mov    eax, [csftdf_w_size]
7008 0000C663 29D2                <1>       sub    edx, edx
7009 0000C665 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
7010 0000C669 01C8                <1>       add    eax, ecx
7011 0000C66B 48                  <1>       dec    eax
7012 0000C66C F7F1                <1>       div    ecx
7013 0000C66E 89C1                <1>       mov    ecx, eax ; sector count
7014 0000C670 A1[D8940100]        <1>       mov    eax, [csftdf_df_cluster]
7015                              <1>
7016                              <1>       ; EBX = memory block address (current)
7017                              <1>
7018 0000C675 E8A20F0000          <1>       call   write_fat_file_sectors
7019 0000C67A 7259                <1>       jc     short csftdf2_write_fat_file_secs_4
7020                              <1>
7021                              <1>       ; EBX = next memory address
7022                              <1>
7023 0000C67C A1[E8940100]        <1>       mov    eax, [csftdf_df_wbytes]
7024 0000C681 0305[E0940100]      <1>       add    eax, [csftdf_w_size]
7025 0000C687 8B15[C8940100]      <1>       mov    edx, [csftdf_filesize]
7026 0000C68D 39D0                <1>       cmp    eax, edx
7027 0000C68F 7344                <1>       jnb    short csftdf2_write_fat_file_secs_4
7028 0000C691 A3[E8940100]        <1>       mov    [csftdf_df_wbytes], eax
7029                              <1>       ;
7030 0000C696 A3[8A940100]        <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
7031                              <1>
7032 0000C69B 53                  <1>       push   ebx ; *
7033                              <1>
7034 0000C69C 803D[C6940100]01    <1>       cmp    byte [DestinationFileFound], 1
7035 0000C6A3 7210                <1>       jb     short csftdf2_write_fat_file_secs_2
7036                              <1>
7037                              <1>       ; get next cluster (csftdf_w_size! bytes)
7038 0000C6A5 A1[D8940100]        <1>       mov    eax, [csftdf_df_cluster]
7039 0000C6AA E887050000          <1>       call   get_next_cluster
7040 0000C6AF 731C                <1>       jnc    short csftdf2_write_fat_file_secs_3
7041                              <1>
7042 0000C6B1 21C0                <1>       and    eax, eax ; end of cluster chain!?
7043 0000C6B3 7521                <1>       jnz    short csftdf2_write_fat_file_secs_5 ; disk error !
7044                              <1>
7045                              <1> csftdf2_write_fat_file_secs_2:
7046 0000C6B5 A1[D8940100]        <1>       mov    eax, [csftdf_df_cluster] ; last cluster
7047 0000C6BA E8800E0000          <1>       call   add_new_cluster
7048 0000C6BF 7215                <1>       jc     short csftdf2_write_fat_file_secs_5
7049                              <1>
7050                              <1>       ; NOTE: Destination file size may be bigger than
7051                              <1>       ; source file size when the last reading fails after here.
7052                              <1>       ; (The last -empty- cluster of destination file must be
7053                              <1>       ; truncated and LMDT must be current date&time for partial
7054                              <1>       ; copy result!)
7055 0000C6C1 8B15[E0940100]      <1>       mov    edx, [csftdf_w_size] ; bytes per cluster
7056 0000C6C7 0115[8A940100]      <1>       add    [DestinationFile_DirEntry+DirEntry_FileSize], edx
7057                              <1>
7058                              <1> csftdf2_write_fat_file_secs_3:
7059 0000C6CD 5B                  <1>       pop    ebx ; *
7060 0000C6CE 29D2                <1>       sub    edx, edx ; 0
7061 0000C6D0 A3[D8940100]        <1>       mov    [csftdf_df_cluster], eax ; next cluster
7062                              <1>
7063                              <1> csftdf2_write_fat_file_secs_4:
7064 0000C6D5 C3                  <1>       retn
7065                              <1>
7066                              <1> csftdf2_write_fat_file_secs_5:
7067 0000C6D6 5B                  <1>       pop    ebx ; *
7068                              <1>       ; 16/10/2016 (1Dh -> 18)
7069 0000C6D7 B812000000          <1>       mov    eax, 18 ; Write error !
7070 0000C6DC C3                  <1>       retn
7071                              <1>
7072                              <1> csftdf2_save_file:
7073                              <1>       ; 09/12/2017
7074                              <1>       ; 25/03/2016
7075                              <1>       ; 19/03/2016
7076                              <1>       ; 18/03/2016
7077 0000C6DD 8B35[F4940100]      <1>       mov    esi, [csftdf_df_drv_dt] ; logical dos drv desc. tbl.
7078                              <1>
7079 0000C6E3 8B1D[CC940100]      <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
7080                              <1>
7081 0000C6E9 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
7082 0000C6ED 0F86F4010000        <1>         jna     csftdf2_save_fs_file
7083                              <1>
7084                              <1> csftdf2_save_fat_file:
7085 0000C6F3 53                  <1>       push   ebx; *
7086                              <1>
7087 0000C6F4 803D[EC940100]00    <1>       cmp    byte [csftdf_percentage], 0
7088 0000C6FB 7724                <1>       ja     short csftdf2_save_fat_file_0
7089                              <1>
7090                              <1>       ; Set cursor position
7091                              <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
7092 0000C6FD 8A3D[ED940100]      <1>       mov    bh, [csftdf_videopage]
7093 0000C703 668B15[EE940100]    <1>       mov    dx, [csftdf_cursorpos]
7094 0000C70A B402                <1>       mov    ah, 2
7095 0000C70C E8F24FFFFF          <1>       call   _int10h
7096                              <1>
7097 0000C711 BE[DF450100]        <1>       mov    esi, msg_writing
7098 0000C716 E80AADFFFF          <1>       call   print_msg
7099                              <1>
7100                              <1> csftdf2_save_fat_file_next:
7101 0000C71B 8B35[F4940100]      <1>       mov    esi, [csftdf_df_drv_dt] ; 25/03/2016
7102                              <1>
7103                              <1> csftdf2_save_fat_file_0:
7104 0000C721 5B                  <1>       pop    ebx ; *
```

```
7105                            <1>
7106                            <1> csftdf2_save_fat_file_1:
7107 0000C722 E813FFFFFF        <1>       call   csftdf2_write_file_sectors ; 19/03/2016
7108 0000C727 0F827E010000      <1>         jc     csftdf2_rw_error ; eocc! or disk error!
7109                            <1>
7110 0000C72D 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
7111 0000C72F 756D              <1>         jnz    short csftdf2_save_fat_file_3 ; 25/03/2016
7112                            <1>
7113 0000C731 803D[EC940100]00  <1>       cmp    byte [csftdf_percentage], 0
7114 0000C738 76E8              <1>       jna    short csftdf2_save_fat_file_1
7115                            <1>
7116 0000C73A B020              <1>       mov    al, 20h
7117 0000C73C BF[EB450100]      <1>       mov    edi, percentagestr
7118 0000C741 AA                <1>       stosb
7119 0000C742 AA                <1>       stosb
7120 0000C743 A1[E8940100]      <1>       mov    eax, [csftdf_df_wbytes]
7121 0000C748 BA64000000        <1>       mov    edx, 100
7122 0000C74D F7E2              <1>       mul    edx
7123 0000C74F 8B0D[C8940100]    <1>       mov    ecx, [csftdf_filesize]
7124 0000C755 F7F1              <1>       div    ecx
7125 0000C757 B10A              <1>       mov    cl, 10
7126 0000C759 F6F1              <1>       div    cl
7127 0000C75B 80C430            <1>       add    ah, '0'
7128 0000C75E 8827              <1>       mov    [edi], ah
7129 0000C760 20C0              <1>       and    al, al
7130 0000C762 740A              <1>       jz     short csftdf2_save_fat_file_2
7131 0000C764 4F                <1>       dec    edi
7132                            <1>       ;cbw
7133 0000C765 30E4              <1>       xor    ah, ah ; 09/12/2017
7134 0000C767 F6F1              <1>       div    cl
7135 0000C769 80C430            <1>       add    ah, '0'
7136 0000C76C 8827              <1>       mov    [edi], ah
7137                            <1>       ;and   al, al
7138                            <1>       ;jz    short csftdf2_save_fat_file_2
7139                            <1>       ;dec   edi
7140                            <1>       ;mov   [edi], '1' ; 100%
7141                            <1>
7142                            <1> csftdf2_save_fat_file_2:
7143 0000C76E 53                <1>       push   ebx ; *
7144                            <1>
7145 0000C76F E802000000        <1>       call   csftdf2_print_wr_percentage ; 25/03/2016
7146                            <1>
7147 0000C774 EBA5              <1>         jmp    csftdf2_save_fat_file_next
7148                            <1>
7149                            <1> csftdf2_print_wr_percentage:
7150                            <1>       ; Set cursor position
7151                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
7152 0000C776 8A3D[ED940100]    <1>       mov    bh, [csftdf_videopage]
7153 0000C77C 668B15[EE940100]  <1>       mov    dx, [csftdf_cursorpos]
7154 0000C783 B402              <1>       mov    ah, 2
7155 0000C785 E8794FFFFF        <1>       call   _int10h
7156                            <1>
7157 0000C78A BE[DF450100]      <1>       mov    esi, msg_writing
7158 0000C78F E891ACFFFF        <1>       call   print_msg
7159                            <1>
7160 0000C794 BE[EB450100]      <1>       mov    esi, percentagestr
7161                            <1>       ;call  print_msg
7162                            <1>       ;retn
7163 0000C799 E987ACFFFF        <1>       jmp    print_msg
7164                            <1>
7165                            <1> csftdf2_save_fat_file_3:
7166 0000C79E 803D[EC940100]00  <1>       cmp    byte [csftdf_percentage], 0
7167 0000C7A5 7611              <1>         jna    csftdf2_save_fat_file_4 ; 25/03/2016
7168                            <1>
7169                            <1>       ; "100%"
7170 0000C7A7 BF[EB450100]      <1>       mov    edi, percentagestr
7171 0000C7AC B031              <1>       mov    al, '1'
7172 0000C7AE AA                <1>       stosb
7173 0000C7AF B030              <1>       mov    al, '0'
7174 0000C7B1 AA                <1>       stosb
7175 0000C7B2 AA                <1>       stosb
7176                            <1>
7177 0000C7B3 E8BEFFFFFF        <1>       call   csftdf2_print_wr_percentage
7178                            <1>
7179                            <1> csftdf2_save_fat_file_4:
7180 0000C7B8 803D[C6940100]00  <1>       cmp    byte [DestinationFileFound], 0
7181 0000C7BF 7647              <1>       jna    short csftdf2_save_fat_file_6
7182                            <1>
7183 0000C7C1 8B35[F4940100]    <1>       mov    esi, [csftdf_df_drv_dt] ; 31/03/2016
7184                            <1>
7185 0000C7C7 A1[D8940100]      <1>       mov    eax, [csftdf_df_cluster] ; last cluster
7186 0000C7CC E865040000        <1>       call   get_next_cluster
7187 0000C7D1 7235              <1>       jc     short csftdf2_save_fat_file_6 ; eocc! or disk error!
7188                            <1>
7189 0000C7D3 A1[D8940100]      <1>       mov    eax, [csftdf_df_cluster] ; last cluster
7190                            <1>       ;xor   ecx, ecx
7191                            <1>       ;mov   [FAT_ClusterCounter], ecx ; 0 ; reset
7192                            <1>       ;dec   ecx ; 0FFFFFFFFh
7193                            <1>       ;shr   ecx, 4 ; 28 bit ; 0FFFFFFFh
7194 0000C7D8 B9FFFFFF0F        <1>       mov    ecx, 0FFFFFFFh
7195 0000C7DD E87E070000        <1>       call   update_cluster
7196 0000C7E2 7224              <1>       jc     short csftdf2_save_fat_file_6 ; really last cluster!?
7197                            <1>
7198 0000C7E4 A3[D8940100]      <1>       mov    [csftdf_df_cluster], eax ; next cluster
7199                            <1>
7200                            <1>       ; byte [FAT_BuffValidData] = 2
7201 0000C7E9 E82F0A0000        <1>       call   save_fat_buffer
7202 0000C7EE 730E              <1>       jnc    short csftdf2_save_fat_file_5
7203                            <1>
7204 0000C7F0 8B15[C8940100]    <1>       mov    edx, [csftdf_filesize]
7205 0000C7F6 8915[8A940100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
7206 0000C7FC EB58              <1>       jmp    short csftdf2_save_fat_file_err3
7207                            <1>
7208                            <1> csftdf2_save_fat_file_5:
7209 0000C7FE A1[D8940100]      <1>       mov    eax, [csftdf_df_cluster]
```

```
                                <1>
                                <1>       ; EAX = First cluster to be truncated/unlinked
                                <1>       ; ESI = Logical dos drive description table address
0000C803 E8580C0000             <1>       call   truncate_cluster_chain
                                <1>
                                <1> csftdf2_save_fat_file_6:
                                <1>       ; 28/03/2016
0000C808 BE[F9930100]           <1>       mov    esi, SourceFile_DirEntry+DirEntry_Attr ; +11 to + 18
0000C80D BF[79940100]           <1>       mov    edi, DestinationFile_DirEntry+DirEntry_Attr ; +11 to + 18
0000C812 A4                     <1>       movsb ; +11
0000C813 A5                     <1>       movsd ; +12 .. +15
0000C814 66A5                   <1>       movsw ; +16 .. +17
                                <1>             ; + 18
0000C816 83C604                 <1>       add    esi, 4
0000C819 83C704                 <1>       add    edi, 4
0000C81C A5                     <1>       movsd  ; DirEntry_WrtTime ; +22 .. +25
                                <1>
0000C81D 8B15[C8940100]         <1>       mov    edx, [csftdf_filesize]
0000C823 8915[8A940100]         <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
                                <1>
0000C829 E8BAF0FFFF             <1>       call   convert_current_date_time
                                <1>       ; DX = Date in dos dir entry format
                                <1>       ; AX = Time in dos dir entry format
0000C82E EB4D                   <1>       jmp    short csftdf2_save_fat_file_7
                                <1>
                                <1> csftdf2_save_fat_file_err1:
0000C830 5B                     <1>       pop    ebx ; *
                                <1> csftdf2_save_fat_file_err2:
0000C831 A1[E8940100]           <1>       mov    eax, [csftdf_df_wbytes]
0000C836 8B15[8A940100]         <1>       mov    edx, [DestinationFile_DirEntry+DirEntry_FileSize]
0000C83C 39C2                   <1>       cmp    edx, eax
0000C83E 7616                   <1>       jna    short csftdf2_save_fat_file_err3
0000C840 A1[D8940100]           <1>       mov    eax, [csftdf_df_cluster] ; last (empty) cluster
0000C845 E8160C0000             <1>       ; ESI = Logical dos drive description table address
0000C84A 720A                   <1>       call   truncate_cluster_chain
0000C84C A1[E8940100]           <1>       jc     short csftdf2_save_fat_file_err3
0000C851 A3[8A940100]           <1>       mov    eax, [csftdf_df_wbytes]
                                <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
                                <1> csftdf2_save_fat_file_err3:
0000C856 E88DF0FFFF             <1>       call   convert_current_date_time
                                <1>       ; DX = Date in dos dir entry format
                                <1>       ; AX = Time in dos dir entry format
0000C85B C605[7B940100]00       <1>       mov    byte [DestinationFile_DirEntry+DirEntry_CrtTimeTenth], 0
0000C862 66A3[7C940100]         <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtTime], ax
0000C868 668915[7E940100]       <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtDate], dx
0000C86F 66A3[84940100]         <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtTime], ax
0000C875 668915[86940100]       <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtDate], dx
0000C87C F9                     <1>       stc
                                <1> csftdf2_save_fat_file_7:
0000C87D 9C                     <1>       pushf
0000C87E 668915[80940100]       <1>       mov    [DestinationFile_DirEntry+DirEntry_LastAccDate], dx
0000C885 BE[6E940100]           <1>       mov    esi, DestinationFile_DirEntry
0000C88A BF00000800             <1>       mov    edi, Directory_Buffer
0000C88F 0FB70D[96940100]       <1>       movzx  ecx, word [DestinationFile_DirEntryNumber] ; (<2048)
0000C896 66C1E105               <1>       shl    cx, 5 ; 32 * directory entry number
0000C89A 01CF                   <1>       add    edi, ecx
                                <1>       ;mov   ecx, 8
0000C89C 66B90800               <1>       mov    cx, 8
0000C8A0 F3A5                   <1>       rep    movsd
0000C8A2 9D                     <1>       popf
0000C8A3 730B                   <1>       jnc    short csftdf2_write_file_OK
                                <1>
                                <1> csftdf2_write_error:
                                <1>       ; 18/03/2016
0000C8A5 B01D                   <1>       mov    al, 1Dh ; write error
0000C8A7 EB02                   <1>       jmp    short csftdf2_rw_error
                                <1>
                                <1>       ; 16/03/2016
                                <1> csftdf2_read_error:
0000C8A9 B011                   <1>       mov    al, 17 ; ; Drive not ready or read error!
                                <1> csftdf2_rw_error:
0000C8AB A2[C5940100]           <1>       mov    [csftdf_rw_err], al
                                <1>
                                <1> csftdf2_write_file_OK:
                                <1>       ; 18/03/2016
0000C8B0 C605[DC900100]02       <1>       mov    byte [DirBuff_ValidData], 2
0000C8B7 E8CAF0FFFF             <1>       call   save_directory_buffer
                                <1>
                                <1>       ; Update last modification date&time of destination
                                <1>       ; file's (parent) directory
0000C8BC E860F1FFFF             <1>       call   update_parent_dir_lmdt
                                <1>       ;
0000C8C1 A1[CC940100]           <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
                                <1>
0000C8C6 21C0                   <1>       and    eax, eax
0000C8C8 750E                   <1>       jnz    short csftdf2_dealloc_mblock
                                <1>
0000C8CA 88C5                   <1>       mov    ch, al ; 0 (Cluster r/w, not full loading)
                                <1> csftdf2_dealloc_retn:
0000C8CC 8A0D[C5940100]         <1>       mov    cl, [csftdf_rw_err]
0000C8D2 A1[D8940100]           <1>       mov    eax, [csftdf_df_cluster]
0000C8D7 C3                     <1>       retn
                                <1>
                                <1> csftdf2_dealloc_mblock:
0000C8D8 8B0D[D0940100]         <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
0000C8DE E87B9CFFFF             <1>       call   deallocate_memory_block
0000C8E3 B5FF                   <1>         mov     ch, 0FFh ; (File was full loaded at memory)
0000C8E5 EBE5                   <1>       jmp    short csftdf2_dealloc_retn
                                <1>
                                <1> csftdf2_save_fs_file:
                                <1>       ; 16/10/2016 (1Dh -> 18)
                                <1>       ; temporary - (21/03/2016)
0000C8E7 B812000000             <1>       mov    eax, 18 ; write error
0000C8EC F9                     <1>       stc
0000C8ED C3                     <1>       retn
```

```
                                <1>
                                <1> create_file:
                                <1>       ; 16/10/2016
                                <1>       ; 24/03/2016, 31/03/2016
                                <1>       ; 20/03/2016, 21/03/2016, 23/03/2016
                                <1>       ; 19/03/2016 (TRDOS 396 = TRDOS v2.0)
                                <1>       ; 03/09/2011 (FILE.ASM, 'proc_create_file')
                                <1>       ; 09/08/2010
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EAX = File Size
                                <1>       ;     ESI = ASCIIZ File Name
                                <1>       ;     CL = File Attributes
                                <1>       ;     EBX = FFFFFFFFh -> create empty file
                                <1>       ;                        (only for FAT fs)
                                <1>       ; OUTPUT ->
                                <1>       ;     CF = 0 ->
                                <1>       ;     EAX = New file's first cluster
                                <1>       ;     ESI = Logical Dos Drv Descr. Table Addr.
                                <1>       ;     EBX = offset CreateFile_Size
                                <1>       ;     ECX = Sectors per cluster (<256)
                                <1>       ;     EDX = Directory entry index/number (<65536)
                                <1>       ;     CF = 1 -> error code in AL
                                <1>
                                <1> ;     test   cl, 18h (directory or volume name)
                                <1> ;     jnz    short loc_createfile_access_denied
0000C8EE 80E107                 <1>       and    cl, 07h ; S, H, R
0000C8F1 880D[14950100]         <1>         mov     [createfile_attrib], cl
                                <1>
0000C8F7 89D9                   <1>       mov    ecx, ebx
0000C8F9 89F3                   <1>       mov    ebx, esi ; ASCIIZ File Name address
0000C8FB 29D2                   <1>       sub    edx, edx
0000C8FD 8A35[B6890100]         <1>         mov     dh, [Current_Drv]
0000C903 BE00010900             <1>         mov     esi, Logical_DOSDisks
0000C908 01D6                   <1>       add    esi, edx
                                <1>
0000C90A 8815[1F950100]         <1>       mov    [createfile_UpdatePDir], dl ; 0 ; 31/03/2016
                                <1>
                                <1>       ; LD_DiskType = 0 for write protection (read only)
0000C910 807E0101               <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
0000C914 730A                   <1>       jnb    short loc_createfile_check_file_sytem
                                <1>       ; 16/10/2016 (TRDOS Error code: 30, disk write protected)
0000C916 B81E000000             <1>       mov    eax, 30 ; 13h, MSDOS err : Disk write-protected
0000C91B 66BA0000               <1>       mov    dx, 0
                                <1>       ; err retn: EDX = 0, EBX = File name offset
                                <1>       ; ESI -> Dos drive description table address
0000C91F C3                     <1>       retn
                                <1>
                                <1> ;loc_createfile_access_denied:
                                <1> ;     mov    eax, 05h ; access denied (invalid attributes input)
                                <1> ;     stc
                                <1> ;     retn
                                <1>
                                <1> loc_createfile_check_file_sytem:
0000C920 807E0301               <1>       cmp    byte [esi+LD_FATType], 1
0000C924 730A                   <1>       jnb    short loc_createfile_chk_empty_FAT_file_sign1
                                <1>
0000C926 A3[00950100]           <1>       mov    [createfile_size], eax
                                <1>       ; ESI = Logical Dos Drive Description Table address
                                <1>       ; EBX = ASCIIZ File Name address
0000C92B E9FE020000             <1>       jmp    create_fs_file
                                <1>
                                <1> loc_createfile_chk_empty_FAT_file_sign1:
                                <1>       ; ECX = FFFFFFFFh -> create empty file if drive has FAT fs
0000C930 41                     <1>       inc    ecx
0000C931 7506                   <1>       jnz    short loc_createfile_chk_empty_FAT_file_sign2
0000C933 890D[00950100]         <1>       mov    [createfile_size], ecx ; 0 ; empty file
                                <1>
                                <1> loc_createfile_chk_empty_FAT_file_sign2:
                                <1>       ; 23/03/2016
0000C939 668B4E11               <1>       mov    cx, [esi+LD_BPB+BytesPerSec]
0000C93D 66890D[1C950100]       <1>       mov    [createfile_BytesPerSec], cx
                                <1>
                                <1>       ; EBX = ASCIIZ File Name address
0000C944 0FB65613               <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
0000C948 8815[15950100]         <1>       mov    [createfile_SecPerClust], dl
0000C94E 8B4E74                 <1>       mov    ecx, [esi+LD_FreeSectors]
0000C951 39D1                   <1>       cmp    ecx, edx ; byte [createfile_SecPerClust]
0000C953 7306                   <1>       jnb    short loc_create_fat_file
                                <1>
                                <1> loc_createfile_insufficient_disk_space:
0000C955 B827000000             <1>       mov    eax, 27h
                                <1> loc_createfile_gffc_retn:
0000C95A C3                     <1>       retn
                                <1>
                                <1> loc_create_fat_file:
0000C95B 891D[F8940100]         <1>       mov    [createfile_Name_Offset], ebx
0000C961 890D[FC940100]         <1>       mov    [createfile_FreeSectors], ecx
                                <1>
                                <1> loc_createfile_gffc_1:
0000C967 E821050000             <1>       call   get_first_free_cluster
0000C96C 72EC                   <1>       jc     short loc_createfile_gffc_retn
                                <1>
0000C96E A3[04950100]           <1>       mov    [createfile_FFCluster], eax
                                <1>
                                <1> loc_createfile_locate_ffe_on_directory:
                                <1>       ; Current directory fcluster <> Directory buffer cluster
                                <1>       ; Current directory will be reloaded by
                                <1>       ; 'locate_current_dir_file' procedure
                                <1>       ;
                                <1>       ; ESI = Logical Dos Drv Desc. Table Adress
0000C973 56                     <1>       push   esi ; *
0000C974 31C0                   <1>       xor    eax, eax
                                <1>
0000C976 A3[D2900100]           <1>       mov    dword [FAT_ClusterCounter], eax ; 0
```

```
7420                                <1>       ; 21/03/2016
7421 0000C97B A2[1E950100]          <1>       mov    byte [createfile_wfc], al ; 0
7422                                <1>
7423 0000C980 89C1                  <1>       mov    ecx, eax
7424 0000C982 6649                  <1>       dec    cx ; FFFFh
7425                                <1>       ; CX = FFFFh -> find first deleted or free entry
7426                                <1>       ; ESI would be ASCIIZ filename address if the call
7427                                <1>       ; would not be for first free or deleted dir entry
7428 0000C984 E8D6E7FFFF            <1>       call   locate_current_dir_file
7429 0000C989 0F83EE000000          <1>       jnc    loc_createfile_set_ff_dir_entry
7430 0000C98F 5E                    <1>       pop    esi ; *
7431                                <1>        ; ESI = Logical DOS Drv. Description Table Address
7432 0000C990 83F802                <1>       cmp    eax, 2
7433 0000C993 7402                  <1>       je     short loc_createfile_add_new_cluster
7434                                <1> loc_createfile_locate_file_stc_retn:
7435 0000C995 F9                    <1>       stc
7436 0000C996 C3                    <1>       retn
7437                                <1>
7438                                <1> loc_createfile_add_new_cluster:
7439 0000C997 803D[B5890100]02      <1>       cmp    byte [Current_FATType], 2
7440                                <1>       ;cmp   byte [esi+LD_FATType], 2
7441 0000C99E 770C                  <1>       ja     short loc_createfile_add_new_cluster_check_fsc
7442 0000C9A0 803D[B4890100]01      <1>       cmp    byte [Current_Dir_Level], 1
7443                                <1>       ;cmp   byte [esi+LD_CDirLevel], 1
7444 0000C9A7 7303                  <1>       jnb    short loc_createfile_add_new_cluster_check_fsc
7445                                <1>
7446                                <1>       ;mov   eax, 12
7447 0000C9A9 B00C                  <1>       mov    al, 12 ; No more files
7448                                <1>
7449                                <1> loc_createfile_anc_retn:
7450 0000C9AB C3                    <1>       retn
7451                                <1>
7452                                <1> loc_createfile_add_new_cluster_check_fsc:
7453 0000C9AC 8B0D[FC940100]        <1>       mov    ecx, [createfile_FreeSectors]
7454 0000C9B2 0FB605[15950100]      <1>       movzx  eax, byte [createfile_SecPerClust]
7455 0000C9B9 66D1E0                <1>       shl    ax, 1 ; AX = 2 * AX
7456 0000C9BC 39C1                  <1>       cmp    ecx, eax
7457 0000C9BE 7295                  <1>         jb   short loc_createfile_insufficient_disk_space
7458                                <1>
7459                                <1> loc_createfile_add_new_subdir_cluster:
7460 0000C9C0 8B15[E1900100]        <1>       mov    edx, [DirBuff_Cluster]
7461 0000C9C6 8915[08950100]        <1>       mov    [createfile_LastDirCluster], edx
7462                                <1>
7463 0000C9CC A1[04950100]          <1>       mov    eax, [createfile_FFCluster]
7464 0000C9D1 E846040000            <1>       call   load_FAT_sub_directory
7465 0000C9D6 72D3                  <1>       jc     short loc_createfile_anc_retn
7466                                <1>
7467                                <1> pass_createfile_add_new_subdir_cluster:
7468                                <1>       ;movzx eax, word [esi+LD_BPB+BytesPerSec]
7469 0000C9D8 0FB705[1C950100]      <1>       movzx  eax, word [createfile_BytesPerSec] ; 23/03/2016
7470 0000C9DF F7E1                  <1>       mul    ecx ; ecx = directory buffer sector count
7471 0000C9E1 89C1                  <1>       mov    ecx, eax
7472 0000C9E3 C1E902                <1>       shr    ecx, 2 ; dword count
7473 0000C9E6 29C0                  <1>       sub    eax, eax ; 0
7474 0000C9E8 F3AB                  <1>       rep    stosd
7475                                <1>       ;
7476 0000C9EA C605[DC900100]02      <1>       mov    byte [DirBuff_ValidData], 2
7477 0000C9F1 E890EFFFFF            <1>       call   save_directory_buffer
7478 0000C9F6 72B3                  <1>       jc     short loc_createfile_anc_retn
7479                                <1>
7480                                <1> loc_createfile_save_added_subdir_cluster:
7481 0000C9F8 A1[08950100]          <1>       mov    eax, [createfile_LastDirCluster]
7482 0000C9FD 8B0D[04950100]        <1>       mov    ecx, [createfile_FFCluster]
7483 0000CA03 E858050000            <1>       call   update_cluster
7484 0000CA08 7304                  <1>       jnc    short loc_createfile_save_fat_buffer_0
7485 0000CA0A 09C0                  <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
7486 0000CA0C 751A                  <1>       jnz    short loc_createfile_save_fat_buffer_stc_retn
7487                                <1>
7488                                <1> loc_createfile_save_fat_buffer_0:
7489 0000CA0E A1[04950100]          <1>       mov    eax, [createfile_FFCluster]
7490 0000CA13 A3[08950100]          <1>       mov    [createfile_LastDirCluster], eax
7491 0000CA18 B9FFFFFF0F            <1>       mov    ecx, 0FFFFFFFh ; 28 bit
7492 0000CA1D E83E050000            <1>       call   update_cluster
7493 0000CA22 7306                  <1>       jnc    short loc_createfile_save_fat_buffer_1
7494 0000CA24 09C0                  <1>       or     eax, eax ; Was it free cluster
7495 0000CA26 7402                  <1>       jz     short loc_createfile_save_fat_buffer_1
7496                                <1>
7497                                <1> loc_createfile_save_fat_buffer_stc_retn:
7498 0000CA28 F9                    <1>       stc
7499                                <1> loc_createfile_save_fat_buffer_retn:
7500                                <1> loc_createfile_gffc_2_stc_retn:
7501 0000CA29 C3                    <1>       retn
7502                                <1>
7503                                <1> loc_createfile_save_fat_buffer_1:
7504                                <1>       ; byte [FAT_BuffValidData] = 2
7505 0000CA2A E8EE070000            <1>       call   save_fat_buffer
7506 0000CA2F 72F8                  <1>       jc     short loc_createfile_save_fat_buffer_retn
7507                                <1>
7508 0000CA31 803D[D2900100]01      <1>       cmp    byte [FAT_ClusterCounter], 1
7509 0000CA38 7222                  <1>       jb     short loc_createfile_save_fat_buffer_2
7510                                <1>
7511                                <1>       ; ESI = Logical DOS Drive Description Table address
7512 0000CA3A A1[D2900100]          <1>       mov    eax, [FAT_ClusterCounter]
7513                                <1>
7514 0000CA3F C605[D2900100]00      <1>       mov    byte [FAT_ClusterCounter], 0 ; 21/03/2016
7515                                <1>
7516 0000CA46 66BB01FF              <1>       mov    bx, 0FF01h ; add free clusters
7517 0000CA4A E863080000            <1>       call   calculate_fat_freespace
7518                                <1>
7519                                <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
7520                                <1>       ;jnz   short loc_createfile_save_fat_buffer_2
7521                                <1>
7522                                <1>       ; ecx > 0 -> Recalculation is needed
7523 0000CA4F 09C9                  <1>       or     ecx, ecx
7524 0000CA51 7409                  <1>       jz     short loc_createfile_save_fat_buffer_2
```

```
7525                                <1>
7526 0000CA53 66BB00FF              <1>       mov    bx, 0FF00h ; ; recalculate free space
7527 0000CA57 E856080000            <1>       call   calculate_fat_freespace
7528                                <1>
7529                                <1> loc_createfile_save_fat_buffer_2:
7530                                <1>       ;call  update_parent_dir_lmdt
7531                                <1>
7532                                <1> loc_createfile_gffc_2:
7533 0000CA5C E82C040000            <1>       call   get_first_free_cluster
7534 0000CA61 72C6                  <1>       jc     short loc_createfile_gffc_2_stc_retn
7535                                <1>
7536 0000CA63 A3[04950100]          <1>       mov    [createfile_FFCluster], eax
7537                                <1>
7538 0000CA68 A1[08950100]          <1>       mov    eax, [createfile_LastDirCluster]
7539                                <1>
7540 0000CA6D E8AA030000            <1>       call   load_FAT_sub_directory
7541 0000CA72 72B5                  <1>       jc     short loc_createfile_gffc_2_stc_retn
7542                                <1>
7543 0000CA74 BF00000800            <1>       mov    edi, Directory_Buffer
7544                                <1>
7545 0000CA79 6629DB                <1>       sub    bx, bx ; directory entry index/number = 0
7546                                <1>
7547 0000CA7C 56                    <1>       push   esi ; * ; 23/03/2016
7548                                <1>
7549                                <1> loc_createfile_set_ff_dir_entry:
7550 0000CA7D 66891D[16950100]      <1>       mov    [createfile_DirIndex], bx
7551                                <1>
7552                                <1>         ; EDI = Directory entry address
7553 0000CA84 8B35[F8940100]        <1>       mov    esi, [createfile_Name_Offset]
7554 0000CA8A A1[04950100]          <1>       mov    eax, [createfile_FFCluster]
7555 0000CA8F A3[0C950100]          <1>       mov    [createfile_Cluster], eax ; 24/03/2016
7556 0000CA94 B5FF                  <1>       mov    ch, 0FFh
7557 0000CA96 8A0D[14950100]        <1>         mov  cl, [createfile_attrib] ; file attributes
7558                                <1>       ; CH > 0 -> File size is in [EBX]
7559 0000CA9C BB[00950100]          <1>       mov    ebx, createfile_size
7560                                <1>
7561 0000CAA1 E803EEFFFF            <1>       call   make_directory_entry
7562                                <1>
7563 0000CAA6 5E                    <1>       pop    esi ; * ; ESI = Logical Dos Drv Desc. Table address
7564                                <1>
7565 0000CAA7 C605[DC900100]02      <1>       mov    byte [DirBuff_ValidData], 2
7566 0000CAAE E8D3EEFFFF            <1>       call   save_directory_buffer
7567 0000CAB3 7221                  <1>       jc     short loc_createfile_set_ff_dir_entry_retn
7568                                <1>
7569 0000CAB5 C605[1F950100]01      <1>       mov    byte [createfile_UpdatePDir], 1 ; 31/03/2016
7570                                <1>
7571                                <1> loc_createfile_get_set_write_file_cluster:
7572 0000CABC A1[00950100]          <1>       mov    eax, [createfile_size]
7573 0000CAC1 09C0                  <1>       or     eax, eax
7574 0000CAC3 7570                  <1>       jnz    short loc_createfile_get_set_wfc_cont
7575 0000CAC5 40                    <1>       inc    eax
7576                                <1>       ; 23/03/2016
7577 0000CAC6 0FB61D[15950100]      <1>       movzx  ebx, byte [createfile_SecPerClust]
7578                                <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
7579 0000CACD 0FB70D[1C950100]      <1>         movzx   ecx, word [createfile_BytesPerSec] ; 512
7580 0000CAD4 EB7C                  <1>       jmp    loc_createfile_set_cluster_count
7581                                <1>
7582                                <1> loc_createfile_set_ff_dir_entry_retn:
7583 0000CAD6 C3                    <1>       retn
7584                                <1>
7585                                <1> loc_createfile_write_fcluster_to_disk:
7586 0000CAD7 034668                <1>       add    eax, [esi+LD_DATABegin] ; convert to physical address
7587 0000CADA BB00000700            <1>       mov    ebx, Cluster_Buffer
7588                                <1>       ; ESI = Logical DOS Drv. Desc. Tbl. address
7589                                <1>       ; EAX = Disk address
7590                                <1>       ; EBX = Sector Buffer
7591                                <1>       ; ECX = sectors per cluster
7592 0000CADF E88A5F0000            <1>       call   disk_write
7593 0000CAE4 7211                  <1>       jc     short loc_createfile_dsk_wr_err
7594                                <1>
7595                                <1> loc_createfile_update_fat_cluster:
7596                                <1>       ; 21/03/2016
7597 0000CAE6 803D[1E950100]00      <1>       cmp    byte [createfile_wfc], 0
7598 0000CAED 7712                  <1>       ja     short loc_createfile_update_fat_cluster_n1
7599                                <1>
7600 0000CAEF FE05[1E950100]        <1>       inc    byte [createfile_wfc] ; 1
7601 0000CAF5 EB24                  <1>       jmp    short loc_createfile_update_fat_cluster_n2
7602                                <1>
7603                                <1> loc_createfile_dsk_wr_err:
7604                                <1>       ; 16/10/2016 (1Dh -> 18)
7605                                <1>       ; 23/03/2016
7606 0000CAF7 B812000000            <1>       mov    eax, 18 ; Drive not ready or write error !
7607 0000CAFC E9BD000000            <1>       jmp    loc_createfile_stc_retn
7608                                <1>
7609                                <1> loc_createfile_update_fat_cluster_n1:
7610 0000CB01 A1[10950100]          <1>       mov    eax, [createfile_PCluster]
7611 0000CB06 8B0D[0C950100]        <1>       mov    ecx, [createfile_Cluster]
7612 0000CB0C E84F040000            <1>       call   update_cluster
7613 0000CB11 7308                  <1>       jnc    short loc_createfile_update_fat_cluster_n2
7614 0000CB13 09C0                  <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
7615 0000CB15 0F85A3000000          <1>       jnz    loc_createfile_stc_retn
7616                                <1>
7617                                <1> loc_createfile_update_fat_cluster_n2:
7618 0000CB1B A1[0C950100]          <1>         mov  eax, [createfile_Cluster]
7619 0000CB20 B9FFFFFF0F            <1>       mov    ecx, 0FFFFFFFh
7620 0000CB25 E836040000            <1>       call   update_cluster
7621 0000CB2A 734E                  <1>       jnc    short loc_createfile_save_fat_buffer_3
7622 0000CB2C 09C0                  <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
7623 0000CB2E 744A                  <1>       jz     short loc_createfile_save_fat_buffer_3
7624                                <1>
7625                                <1> loc_createfile_upd_fat_fcluster_stc_retn:
7626 0000CB30 E989000000            <1>       jmp    loc_createfile_stc_retn
7627                                <1>
7628                                <1> loc_createfile_get_set_wfc_cont:
7629                                <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
```

```
7630 0000CB35 0FB70D[1C950100]    <1>       movzx  ecx, word [createfile_BytesPerSec] ; 512
7631 0000CB3C 01C8                <1>       add    eax, ecx
7632 0000CB3E 48                  <1>       dec    eax  ; add eax, 511
7633 0000CB3F 29D2                <1>       sub    edx, edx
7634 0000CB41 F7F1                <1>       div    ecx
7635 0000CB43 0FB61D[15950100]    <1>       movzx  ebx, byte [createfile_SecPerClust]
7636 0000CB4A 01D8                <1>       add    eax, ebx
7637 0000CB4C 48                  <1>       dec    eax  ; add eax, SecPerClust - 1
7638 0000CB4D 6631D2              <1>       xor    dx, dx
7639 0000CB50 F7F3                <1>       div    ebx
7640                              <1>
7641                              <1> loc_createfile_set_cluster_count:
7642 0000CB52 A3[18950100]        <1>       mov    [createfile_CCount], eax
7643                              <1>
7644 0000CB57 BF00000700          <1>       mov    edi, Cluster_Buffer
7645 0000CB5C 89C8                <1>       mov    eax, ecx ; Bytes per Sector
7646 0000CB5E F7E3                <1>       mul    ebx ; Sectors per Cluster
7647                              <1>       ; EAX = Bytes per Cluster
7648 0000CB60 89C1                <1>       mov    ecx, eax
7649 0000CB62 C1E902              <1>       shr    ecx, 2 ; dword count
7650 0000CB65 31C0                <1>       xor    eax, eax
7651 0000CB67 F3AB                <1>       rep    stosd ; clear cluster buffer
7652                              <1>
7653 0000CB69 A1[0C950100]        <1>       mov    eax, [createfile_Cluster] ; 24/03/2016
7654                              <1>
7655 0000CB6E 89D9                <1>       mov    ecx, ebx
7656                              <1>
7657                              <1> loc_createfile_get_set_wf_fclust_cont:
7658 0000CB70 83E802              <1>       sub    eax, 2
7659 0000CB73 F7E1                <1>       mul    ecx
7660                              <1>       ; EAX = Logical DOS disk address (offset)
7661 0000CB75 E95DFFFFFF          <1>         jmp     loc_createfile_write_fcluster_to_disk
7662                              <1>
7663                              <1> loc_createfile_save_fat_buffer_3:
7664                              <1>       ; byte [FAT_BuffValidData] = 2
7665 0000CB7A E89E060000          <1>       call   save_fat_buffer
7666 0000CB7F 723D                <1>       jc     loc_createfile_stc_retn
7667                              <1>
7668                              <1>       ; 21/03/2016
7669 0000CB81 803D[D2900100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
7670 0000CB88 721B                <1>       jb     short loc_createfile_save_fat_buffer_4
7671                              <1>
7672                              <1>       ; ESI = Logical DOS Drive Description Table address
7673 0000CB8A A1[D2900100]        <1>       mov    eax, [FAT_ClusterCounter]
7674 0000CB8F 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
7675 0000CB93 E81A070000          <1>       call   calculate_fat_freespace
7676                              <1>
7677                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
7678                              <1>       ;jnz   short loc_createfile_save_fat_buffer_4
7679                              <1>
7680                              <1>       ; ecx > 0 -> Recalculation is needed
7681 0000CB98 09C9                <1>       or     ecx, ecx
7682 0000CB9A 7409                <1>       jz     short loc_createfile_save_fat_buffer_4
7683                              <1>
7684 0000CB9C 66BB00FF            <1>       mov    bx, 0FF00h ; ; recalculate free space
7685 0000CBA0 E80D070000          <1>       call   calculate_fat_freespace
7686                              <1>
7687                              <1> loc_createfile_save_fat_buffer_4:
7688 0000CBA5 FF0D[18950100]      <1>       dec    dword [createfile_CCount]
7689                              <1>       ;jz    short loc_createfile_upd_dir_modif_date_time
7690 0000CBAB 743F                <1>       jz     short loc_createfile_stc_retn_cc ; 31/03/2016
7691                              <1>
7692                              <1> loc_createfile_get_set_write_next_cluster:
7693 0000CBAD E8DB020000          <1>       call   get_first_free_cluster
7694 0000CBB2 720A                <1>       jc     short loc_createfile_stc_retn
7695                              <1>
7696                              <1> loc_createfile_get_set_write_next_cluster_1:
7697 0000CBB4 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
7698 0000CBB7 7213                <1>       jb     short loc_createfile_get_set_write_next_cluster_2
7699                              <1>
7700                              <1> loc_createfile_wnc_insufficient_disk_space:
7701 0000CBB9 B827000000          <1>       mov    eax, 27h ; Insufficient disk space
7702                              <1>
7703                              <1> loc_createfile_stc_retn:
7704 0000CBBE 803D[1E950100]01    <1>       cmp    byte [createfile_wfc], 1
7705 0000CBC5 7324                <1>       jnb    short loc_createfile_err_retn
7706 0000CBC7 C3                  <1>       retn
7707                              <1>
7708                              <1> loc_createfile_wnc_inv_format_retn:
7709                              <1>       ;mov   eax, 28
7710 0000CBC8 B01C                <1>       mov    al, 28 ; Invalid format
7711 0000CBCA EBF2                <1>       jmp    short loc_createfile_stc_retn
7712                              <1>
7713                              <1> loc_createfile_get_set_write_next_cluster_2:
7714 0000CBCC 83F802              <1>       cmp    eax, 2
7715 0000CBCF 72F7                <1>       jb     short loc_createfile_wnc_inv_format_retn
7716                              <1>
7717                              <1> loc_createfile_get_set_write_next_cluster_3:
7718 0000CBD1 8B0D[0C950100]      <1>       mov    ecx, [createfile_Cluster]
7719 0000CBD7 A3[0C950100]        <1>       mov    [createfile_Cluster], eax
7720 0000CBDC 890D[10950100]      <1>       mov    [createfile_PCluster], ecx
7721 0000CBE2 0FB60D[15950100]    <1>       movzx  ecx, byte [createfile_SecPerClust]
7722 0000CBE9 EB85                <1>       jmp    short loc_createfile_get_set_wf_fclust_cont
7723                              <1>
7724                              <1> loc_createfile_err_retn:
7725 0000CBEB F9                  <1>       stc
7726                              <1>
7727                              <1> ;loc_createfile_upd_dir_modif_date_time:
7728                              <1> loc_createfile_stc_retn_cc: ; 31/03/2016
7729 0000CBEC 9C                  <1>       pushf  ; cpu is here for an error return or completion
7730 0000CBED 50                  <1>       push   eax ; error code if cf = 1
7731                              <1>
7732                              <1>       ;call  update_parent_dir_lmdt
7733                              <1>
7734                              <1> ;loc_createfile_stc_retn_cc:
```

```
7735 0000CBEE A1[D2900100]        <1>       mov    eax, [FAT_ClusterCounter]
7736 0000CBF3 09C0                <1>       or     eax, eax
7737 0000CBF5 741A                <1>       jz     short loc_createfile_stc_retn_pop_eax
7738 0000CBF7 8A3D[B6890100]      <1>       mov    bh, [Current_Drv]
7739 0000CBFD B301                <1>       mov    bl, 01h ; BL = 1 -> add clusters
7740                              <1>       ; NOTE: EAX value will be added to Free Cluster Count
7741                              <1>       ; (If EAX value is negative, Free Cluster Count will be decreased)
7742 0000CBFF E8AE060000          <1>       call   calculate_fat_freespace
7743                              <1>         ; ESI = Logical DOS Drive Description Table Address
7744                              <1>         ;jc  short loc_createfile_stc_retn_pop_eax_cf
7745 0000CC04 21C9                <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
7746 0000CC06 7409                <1>       jz     short loc_createfile_stc_retn_pop_eax
7747                              <1>
7748                              <1> loc_createfile_stc_retn_recalc_FAT_freespace:
7749 0000CC08 66BB00FF            <1>       mov    bx, 0FF00h ; bh = 0FFh ->
7750                              <1>       ; ESI = Logical DOS Drv DT Addr
7751                              <1>       ; BL = 0 -> Recalculate
7752 0000CC0C E8A1060000          <1>       call   calculate_fat_freespace
7753                              <1>
7754                              <1> loc_createfile_stc_retn_pop_eax:
7755 0000CC11 58                  <1>       pop    eax
7756 0000CC12 9D                  <1>       popf
7757 0000CC13 7218                <1>       jc     short loc_createfile_retn
7758                              <1>
7759                              <1> loc_createfile_retn_fcluster:
7760 0000CC15 A1[04950100]        <1>       mov    eax, [createfile_FFCluster]
7761 0000CC1A BB[00950100]        <1>       mov    ebx, createfile_size
7762                              <1>       ;movzx ecx, byte [esi+LD_BPB+SecPerClust]
7763 0000CC1F 0FB60D[15950100]    <1>       movzx  ecx, byte [createfile_SecPerClust] ; 23/03/2016
7764 0000CC26 0FB715[16950100]    <1>       movzx  edx, word [createfile_DirIndex]
7765                              <1>
7766                              <1> loc_createfile_retn:
7767 0000CC2D C3                  <1>       retn
7768                              <1>
7769                              <1> create_fs_file:
7770                              <1>       ; temporary (21/03/2016)
7771 0000CC2E C3                  <1>       retn
7772                              <1>
7773                              <1> delete_fs_file:
7774                              <1>       ; temporary (28/02/2016)
7775 0000CC2F C3                  <1>       retn
7776                              <1>
7777                              <1> rename_fs_file_or_directory:
7778 0000CC30 C3                  <1>       retn
7779                              <1>
7780                              <1> make_fs_directory:
7781                              <1>       ; temporary (21/02/2016)
7782 0000CC31 C3                  <1>       retn
7783                              <1>
7784                              <1> add_new_fs_section:
7785                              <1>       ; temporary (11/03/2016)
7786 0000CC32 C3                  <1>       retn
7787                              <1>
7788                              <1> delete_fs_directory_entry:
7789                              <1>       ; temporary (11/03/2016)
7790 0000CC33 C3                  <1>       retn
7791                              <1>
7792                              <1> csftdf2_read_fs_file_sectors:
7793                              <1>       ; temporary (19/03/2016)
7794 0000CC34 C3                  <1>       retn
7795                              <1>
7796                              <1> csftdf2_write_fs_file_sectors:
7797                              <1>       ; temporary (19/03/2016)
7798 0000CC35 C3                  <1>       retn
3088                                  %include 'trdosk5.s' ; 24/01/2016
3089                              <1> ; ****************************************************************************
3090                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - File System Procedures : trdosk5s
3091                              <1> ; ----------------------------------------------------------------------------
3092                              <1> ; Last Update: 23/10/2016
3093                              <1> ; ----------------------------------------------------------------------------
3094                              <1> ; Beginning: 24/01/2016
3095                              <1> ; ----------------------------------------------------------------------------
3096                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
3097                              <1> ; ----------------------------------------------------------------------------
3098                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3099                              <1> ; DRV_FAT.ASM (21/08/2011)
3100                              <1> ; ****************************************************************************
3101                              <1> ; DRV_FAT.ASM (c) 2005-2011 Erdogan TAN [ 07/07/2009 ] Last Update: 21/08/2011
3102                              <1>
3103                              <1> get_next_cluster:
3104                              <1>       ; 15/10/2016
3105                              <1>       ; 23/03/2016
3106                              <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
3107                              <1>       ; 05/07/2011
3108                              <1>       ; 07/07/2009
3109                              <1>       ; 2005
3110                              <1>       ; INPUT ->
3111                              <1>       ;     EAX = Cluster Number (32 bit)
3112                              <1>       ;     ESI = Logical DOS Drive Parameters Table
3113                              <1>       ; OUTPUT ->
3114                              <1>       ;     cf = 0 -> No Error, EAX valid
3115                              <1>       ;     cf = 1 & EAX = 0 -> End Of Cluster Chain
3116                              <1>       ;     cf = 1 & EAX > 0 -> Error
3117                              <1>       ;     ECX = Current/Previous cluster (if CF = 0)
3118                              <1>       ;     EAX = Next Cluster Number (32 bit)
3119                              <1>       ;
3120                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3121                              <1>
3122 0000CC36 A3[C6900100]        <1>       mov    [FAT_CurrentCluster], eax
3123                              <1> check_next_cluster_fat_type:
3124 0000CC3B 29D2                <1>       sub    edx, edx ; 0
3125 0000CC3D 807E0302            <1>         cmp     byte [esi+LD_FATType], 2
3126 0000CC41 7250                <1>       jb     short get_FAT12_next_cluster
3127 0000CC43 0F87AF000000        <1>         ja      get_FAT32_next_cluster
3128                              <1> get_FAT16_next_cluster:
```

```
3129 0000CC49 BB00030000          <1>       mov    ebx, 300h ;768
3130 0000CC4E F7F3                <1>       div    ebx
3131                              <1>       ; EAX = Count of 3 FAT sectors
3132                              <1>       ; EDX = Cluster Offset (< 768)
3133 0000CC50 66D1E2              <1>       shl    dx, 1 ; Multiply by 2
3134 0000CC53 89D3                <1>       mov    ebx, edx ; Byte Offset
3135 0000CC55 81C3001C0900        <1>       add    ebx, FAT_Buffer
3136 0000CC5B 66BA0300            <1>       mov    dx, 3
3137 0000CC5F F7E2                <1>       mul    edx
3138                              <1>       ; EAX = FAT Sector (<= 256)
3139                              <1>       ; EDX = 0
3140 0000CC61 8A0E                <1>       mov    cl, [esi+LD_Name]
3141 0000CC63 803D[CA900100]00    <1>       cmp    byte [FAT_BuffValidData], 0
3142 0000CC6A 0F86CC000000        <1>         jna     load_FAT_sectors0
3143 0000CC70 3A0D[CB900100]      <1>       cmp    cl, [FAT_BuffDrvName]
3144 0000CC76 0F85C0000000        <1>         jne     load_FAT_sectors0
3145 0000CC7C 3B05[CE900100]      <1>       cmp    eax, [FAT_BuffSector]
3146 0000CC82 0F85BA000000        <1>         jne     load_FAT_sectors1
3147                              <1>       ;movzx eax, word [ebx]
3148 0000CC88 668B03              <1>       mov    ax, [ebx]
3149                              <1>       ; 01/02/2016
3150                              <1>       ; DRV_FAT.ASM (21/08/2011) had a FATal bug here !
3151                              <1>       ; (cmp ah, 0Fh) ! (ax >= FF7h)
3152                              <1>       ; (how can i do a such mistake!?)
3153                              <1>       ;cmp   al, 0F7h
3154                              <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3155                              <1>       ;cmp   ah, 0FFh
3156                              <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3157 0000CC8B 6683F8F7            <1>       cmp    ax, 0FFF7h
3158 0000CC8F 725A                <1>       jb     short loc_pass_gnc_FAT16_eoc_check
3159                              <1>       ; ax >= FFF7h (cluster 0002h to FFF6h is valid, in use)
3160 0000CC91 EB56                <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
3161                              <1>
3162                              <1> get_FAT12_next_cluster:
3163 0000CC93 BB00040000          <1>       mov    ebx, 400h ;1024
3164 0000CC98 F7F3                <1>       div    ebx
3165                              <1>       ; EAX = Count of 3 FAT sectors
3166                              <1>       ; EDX = Cluster Offset (< 1024)
3167 0000CC9A 6650                <1>       push   ax
3168 0000CC9C 66B80300            <1>       mov    ax, 3
3169 0000CCA0 66F7E2              <1>       mul    dx     ; Multiply by 3
3170 0000CCA3 66D1E8              <1>       shr    ax, 1  ; Divide by 2
3171 0000CCA6 6689C3              <1>         mov  bx, ax       ; Byte Offset
3172 0000CCA9 81C3001C0900        <1>       add    ebx, FAT_Buffer
3173 0000CCAF 6658                <1>       pop    ax
3174 0000CCB1 66BA0300            <1>       mov    dx, 3
3175 0000CCB5 F7E2                <1>       mul    edx
3176                              <1>       ; EAX = FAT Sector (<= 12)
3177                              <1>       ; EDX = 0
3178 0000CCB7 8A0E                <1>       mov    cl, [esi+LD_Name]
3179 0000CCB9 803D[CA900100]00    <1>       cmp    byte [FAT_BuffValidData], 0
3180 0000CCC0 767A                <1>       jna    short load_FAT_sectors0
3181 0000CCC2 3A0D[CB900100]      <1>       cmp    cl, [FAT_BuffDrvName]
3182 0000CCC8 7572                <1>       jne    short load_FAT_sectors0
3183 0000CCCA 3B05[CE900100]      <1>       cmp    eax, [FAT_BuffSector]
3184 0000CCD0 7570                <1>       jne    short load_FAT_sectors1
3185 0000CCD2 A1[C6900100]        <1>       mov    eax, [FAT_CurrentCluster]
3186 0000CCD7 66D1E8              <1>       shr    ax, 1
3187                              <1>       ;movzx eax, word [ebx]
3188 0000CCDA 668B03              <1>       mov    ax, [ebx]
3189 0000CCDD 7314                <1>       jnc    short get_FAT12_nc_even
3190 0000CCDF 66C1E804            <1>       shr    ax, 4
3191                              <1> loc_gnc_fat12_eoc_check:
3192                              <1>       ;cmp   al, 0F7h
3193                              <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3194                              <1>       ;cmp   ah, 0Fh
3195                              <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3196 0000CCE3 663DF70F            <1>       cmp    ax, 0FF7h
3197 0000CCE7 7202                <1>       jb     short loc_pass_gnc_FAT16_eoc_check
3198                              <1>       ; ax >= FF7h (cluster 0002h to FF6h is valid, in use)
3199                              <1>
3200                              <1> loc_pass_gnc_FAT16_eoc_check_xor_eax:
3201 0000CCE9 31C0                <1>       xor    eax, eax ; 0
3202                              <1> loc_pass_gnc_FAT16_eoc_check:
3203                              <1> loc_pass_gnc_FAT32_eoc_check:
3204 0000CCEB 8B0D[C6900100]      <1>       mov    ecx, [FAT_CurrentCluster]
3205 0000CCF1 F5                  <1>       cmc
3206 0000CCF2 C3                  <1>       retn
3207                              <1>
3208                              <1> get_FAT12_nc_even:
3209 0000CCF3 80E40F              <1>       and    ah, 0Fh
3210 0000CCF6 EBEB                <1>       jmp    short loc_gnc_fat12_eoc_check
3211                              <1>
3212                              <1> get_FAT32_next_cluster:
3213 0000CCF8 BB80010000          <1>       mov    ebx, 180h ;384
3214 0000CCFD F7F3                <1>       div    ebx
3215                              <1>       ; EAX = Count of 3 FAT sectors
3216                              <1>       ; EDX = Cluster Offset (< 384)
3217 0000CCFF 66C1E202            <1>       shl    dx, 2  ; Multiply by 4
3218 0000CD03 89D3                <1>       mov    ebx, edx ; Byte Offset
3219 0000CD05 81C3001C0900        <1>       add    ebx, FAT_Buffer
3220 0000CD0B 66BA0300            <1>       mov    dx, 3
3221 0000CD0F F7E2                <1>       mul    edx
3222                              <1>         ; EAX = FAT Sector (<= 2097152) ; (FFFFFF7h * 4) / 512
3223                              <1>       ;      for 32KB cluster size:
3224                              <1>       ;      EAX <= 1024 = (4GB / 32KB) * 4) / 512
3225                              <1>       ; EDX = 0
3226 0000CD11 8A0E                <1>       mov    cl, [esi+LD_Name]
3227 0000CD13 803D[CA900100]00    <1>       cmp    byte [FAT_BuffValidData], 0
3228 0000CD1A 7620                <1>       jna    short load_FAT_sectors0
3229 0000CD1C 3A0D[CB900100]      <1>       cmp    cl, [FAT_BuffDrvName]
3230 0000CD22 7518                <1>       jne    short load_FAT_sectors0
3231 0000CD24 3B05[CE900100]      <1>       cmp    eax, [FAT_BuffSector] ; 0, 3, 6, 9 ...
3232 0000CD2A 7516                <1>       jne    short load_FAT_sectors1
3233 0000CD2C 8B03                <1>       mov    eax, [ebx]
```

```
3234 0000CD2E 25FFFFFF0F          <1>       and    eax, 0FFFFFFFh ; 28 bit Cluster
3235 0000CD33 3DF7FFFF0F          <1>       cmp    eax, 0FFFFFF7h
3236 0000CD38 72B1                <1>       jb     short loc_pass_gnc_FAT32_eoc_check
3237                              <1>       ; eax >= FFFFFF7h (cluster 0002h to FFFFFF6h is valid)
3238 0000CD3A EBAD                <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
3239                              <1>
3240                              <1> load_FAT_sectors0:
3241 0000CD3C 880D[CB900100]      <1>       mov    [FAT_BuffDrvName], cl
3242                              <1> load_FAT_sectors1:
3243 0000CD42 A3[CE900100]        <1>       mov    [FAT_BuffSector], eax
3244 0000CD47 89C3                <1>       mov    ebx, eax
3245 0000CD49 034660              <1>         add     eax, [esi+LD_FATBegin]
3246 0000CD4C 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
3247 0000CD50 7706                <1>         ja      short load_FAT_sectors3
3248 0000CD52 0FB74E1C            <1>       movzx  ecx, word [esi+LD_BPB+BPB_FATSz16]
3249 0000CD56 EB03                <1>       jmp    short load_FAT_sectors4
3250                              <1> load_FAT_sectors3:
3251 0000CD58 8B4E2A              <1>       mov    ecx, [esi+LD_BPB+BPB_FATSz32]
3252                              <1> load_FAT_sectors4:
3253 0000CD5B 29D9                <1>       sub    ecx, ebx ; [FAT_BuffSector]
3254 0000CD5D 83F903              <1>         cmp     ecx, 3
3255 0000CD60 7605                <1>         jna     short load_FAT_sectors5
3256 0000CD62 B903000000          <1>       mov    ecx, 3
3257                              <1> load_FAT_sectors5:
3258 0000CD67 BB001C0900          <1>       mov    ebx, FAT_Buffer
3259 0000CD6C E80C5D0000          <1>       call   disk_read
3260 0000CD71 730D                <1>       jnc    short load_FAT_sectors_ok
3261                              <1>       ; 15/10/2016 (15h -> 17)
3262                              <1>       ; 23/03/2016 (15h)
3263 0000CD73 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
3264 0000CD78 C605[CA900100]00    <1>       mov    byte [FAT_BuffValidData], 0
3265 0000CD7F C3                  <1>       retn
3266                              <1> load_FAT_sectors_ok:
3267 0000CD80 C605[CA900100]01    <1>       mov    byte [FAT_BuffValidData], 1
3268 0000CD87 A1[C6900100]        <1>       mov    eax, [FAT_CurrentCluster]
3269 0000CD8C E9AAFEFFFF          <1>         jmp     check_next_cluster_fat_type
3270                              <1>
3271                              <1> load_FAT_root_directory:
3272                              <1>       ; 23/10/2016
3273                              <1>       ; 15/10/2016
3274                              <1>       ; 07/02/2016
3275                              <1>       ; 02/02/2016
3276                              <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
3277                              <1>       ; 21/05/2011
3278                              <1>       ; 22/08/2009
3279                              <1>       ;
3280                              <1>       ; INPUT ->
3281                              <1>       ;     ESI = Logical DOS Drive Description Table
3282                              <1>       ; OUTPUT ->
3283                              <1>       ;     cf = 1 -> Root directory could not be loaded
3284                              <1>       ;         EAX > 0 -> Error number
3285                              <1>       ;     cf = 0 -> EAX = 0
3286                              <1>       ;     ECX = Directory buffer size in sectors (CL)
3287                              <1>       ;     EBX = Directory buffer address
3288                              <1>       ;     NOTE: DirBuffer_Size is in bytes ! (word)
3289                              <1>       ;
3290                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3291                              <1>
3292                              <1>       ; NOTE: Only for FAT12 and FAT16 file systems !
3293                              <1>       ; (FAT32 fs root dir must be loaded as sub directory)
3294                              <1>
3295 0000CD91 8A1E                <1>       mov    bl, [esi+LD_Name]
3296 0000CD93 8A7E03              <1>       mov    bh, [esi+LD_FATType]
3297                              <1>
3298                              <1>       ;mov   [DirBuff_DRV], bl
3299                              <1>       ;mov   [DirBuff_FATType], bh
3300 0000CD96 66891D[DA900100]    <1>       mov    [DirBuff_DRV], bx
3301                              <1>
3302                              <1>       ;cmp   bh, 2
3303                              <1>       ;ja    short load_FAT32_root_dir0 ; FAT32 root dir
3304                              <1>
3305                              <1> load_FAT_root_dir0: ; 23/10/2016
3306 0000CD9D 0FB75617            <1>       movzx  edx, word [esi+LD_BPB+RootDirEnts]
3307                              <1>
3308                              <1>       ;or    dx, dx ; 0 for FAT32 file systems
3309                              <1>       ;jz    short load_FAT32_root_dir0 ; FAT32 root dir
3310                              <1>
3311 0000CDA1 6681FA0002          <1>       cmp    dx, 512 ; Number of Root Dir Entries
3312 0000CDA6 7414                <1>       je     short lrd_mov_ecx_32
3313 0000CDA8 89D0                <1>       mov    eax, edx
3314                              <1>       ; 23/10/2016
3315 0000CDAA 89C1                <1>       mov    ecx, eax
3316 0000CDAC 6683C10F            <1>       add    cx, 15 ; round up
3317 0000CDB0 66C1E904            <1>       shr    cx, 4  ; 16 entries per sector (512/32)
3318                              <1>       ; ecx = Root directory size in sectors
3319 0000CDB4 66C1E005            <1>       shl    ax, 5 ; Root directory size in bytes
3320 0000CDB8 664A                <1>       dec    dx    ; Last entry number of root dir
3321                              <1>       ; cx = Dir Buffer sector count
3322 0000CDBA EB0B                <1>       jmp    short lrd_check_dir_buffer
3323                              <1>
3324                              <1> lrd_mov_ecx_32:
3325 0000CDBC B920000000          <1>       mov    ecx, 32
3326 0000CDC1 664A                <1>       dec    dx ; 511
3327 0000CDC3 66B80040            <1>       mov    ax, 32*512
3328                              <1>
3329                              <1> lrd_check_dir_buffer:
3330 0000CDC7 29DB                <1>       sub    ebx, ebx ; 0
3331 0000CDC9 881D[DC900100]      <1>       mov    [DirBuff_ValidData], bl ; 0
3332 0000CDCF 668915[DF900100]    <1>       mov    [DirBuff_LastEntry], dx
3333 0000CDD6 891D[E1900100]      <1>       mov    [DirBuff_Cluster], ebx ; 0
3334 0000CDDC 66A3[E5900100]      <1>       mov    [DirBuffer_Size], ax
3335                              <1>
3336 0000CDE2 8B4664              <1>       mov    eax, [esi+LD_ROOTBegin]
3337                              <1> read_directory:
3338 0000CDE5 BB00000800          <1>       mov    ebx, Directory_Buffer
```

```
3339 0000CDEA 51                  <1>       push   ecx ; Directory buffer sector count
3340 0000CDEB 53                  <1>       push   ebx
3341 0000CDEC E88C5C0000          <1>       call   disk_read
3342 0000CDF1 5B                  <1>       pop    ebx
3343 0000CDF2 720B                <1>       jc     short load_DirBuff_error
3344                              <1>
3345                              <1> validate_DirBuff_and_return:
3346 0000CDF4 59                  <1>       pop    ecx ; Number of loaded sectors
3347 0000CDF5 C605[DC900100]01    <1>       mov    byte [DirBuff_ValidData], 1
3348 0000CDFC 31C0                <1>       xor    eax, eax ; 0 = no error
3349 0000CDFE C3                  <1>       retn
3350                              <1>
3351                              <1> load_DirBuff_error:
3352 0000CDFF 89C8                <1>       mov    eax, ecx ; remaining sectors
3353 0000CE01 59                  <1>       pop    ecx ; sector count
3354 0000CE02 29C1                <1>       sub    ecx, eax ; Number of loaded sectors
3355                              <1>       ; 15/10/2016 (15h -> 17)
3356 0000CE04 B811000000          <1>       mov    eax, 17 ; DRV NOT READY OR READ ERROR !
3357 0000CE09 F9                  <1>       stc
3358 0000CE0A C3                  <1>         retn
3359                              <1>
3360                              <1> load_FAT32_root_directory:
3361                              <1>       ; 02/02/2016 (TRDOS 386 =  TRDOS v2.0)
3362                              <1>       ;
3363                              <1>       ; INPUT ->
3364                              <1>       ;      ESI = Logical DOS Drive Description Table
3365                              <1>       ; OUTPUT ->
3366                              <1>       ;      cf = 1 -> Root directory could not be loaded
3367                              <1>       ;          EAX > 0 -> Error number
3368                              <1>       ;      cf = 0 -> EAX = 0
3369                              <1>       ;      ECX = Directory buffer size in sectors (CL)
3370                              <1>       ;      EBX = Directory buffer address
3371                              <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
3372                              <1>       ;
3373                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3374                              <1>
3375                              <1>
3376 0000CE0B 8A1E                <1>       mov    bl, [esi+LD_Name]
3377 0000CE0D 8A7E03              <1>       mov    bh, [esi+LD_FATType]
3378                              <1>
3379                              <1>       ;mov   [DirBuff_DRV], bl
3380                              <1>       ;mov   [DirBuff_FATType], bh
3381 0000CE10 66891D[DA900100]    <1>       mov    [DirBuff_DRV], bx
3382                              <1>
3383                              <1> load_FAT32_root_dir0:
3384 0000CE17 8B4632              <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
3385 0000CE1A EB0C                <1>       jmp    short load_FAT_sub_dir0
3386                              <1>
3387                              <1> load_FAT_sub_directory:
3388                              <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
3389                              <1>       ; 05/07/2011
3390                              <1>       ; 23/08/2009
3391                              <1>       ;
3392                              <1>       ; INPUT ->
3393                              <1>       ;      ESI = Logical DOS Drive Description Table
3394                              <1>       ;      EAX = Cluster Number
3395                              <1>       ; OUTPUT ->
3396                              <1>       ;      cf = 1 -> Sub directory could not be loaded
3397                              <1>       ;          EAX > 0 -> Error number
3398                              <1>       ;      cf = 0 -> EAX = 0
3399                              <1>       ;      ECX = Directory buffer size in sectors (CL)
3400                              <1>       ;      EBX = Directory buffer address
3401                              <1>       ;
3402                              <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
3403                              <1>       ;
3404                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3405                              <1>
3406 0000CE1C 8A1E                <1>       mov    bl, [esi+LD_Name]
3407 0000CE1E 8A7E03              <1>       mov    bh, [esi+LD_FATType]
3408                              <1>
3409                              <1>       ;mov   [DirBuff_DRV], bl
3410                              <1>       ;mov   [DirBuff_FATType], bh
3411 0000CE21 66891D[DA900100]    <1>       mov    [DirBuff_DRV], bx
3412                              <1>
3413                              <1> load_FAT_sub_dir0:
3414 0000CE28 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
3415                              <1>
3416 0000CE2C 882D[DC900100]      <1>       mov    [DirBuff_ValidData], ch ; 0
3417 0000CE32 A3[E1900100]        <1>       mov    [DirBuff_Cluster], eax
3418                              <1>
3419 0000CE37 0FB74611            <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
3420 0000CE3B F7E1                <1>       mul    ecx
3421 0000CE3D C1E805              <1>       shr    eax, 5 ; directory entry count (dir size / 32)
3422 0000CE40 6648                <1>       dec    ax ; last entry
3423 0000CE42 66A3[DF900100]      <1>       mov    [DirBuff_LastEntry], ax
3424                              <1>
3425 0000CE48 A1[E1900100]        <1>       mov    eax, [DirBuff_Cluster]
3426 0000CE4D 83E802              <1>       sub    eax, 2
3427 0000CE50 F7E1                <1>       mul    ecx
3428 0000CE52 034668              <1>       add    eax, [esi+LD_DATABegin]
3429                              <1>       ; ecx = sector per cluster (dir buffer size = 32 sectors)
3430 0000CE55 EB8E                <1>       jmp    short read_directory
3431                              <1>
3432                              <1> ; DRV_FS.ASM
3433                              <1>
3434                              <1> load_current_FS_directory:
3435 0000CE57 C3                  <1>       retn
3436                              <1> load_FS_root_directory:
3437 0000CE58 C3                  <1>       retn
3438                              <1> load_FS_sub_directory:
3439 0000CE59 C3                  <1>       retn
3440                              <1>
3441                              <1> read_cluster:
3442                              <1>       ; 15/10/2016
3443                              <1>       ; 18/03/2016
```

```
                                <1>       ; 16/03/2016
                                <1>       ; 17/02/2016
                                <1>       ; 15/02/2016 (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 loaded at the buffer
                                <1>       ;         EAX > 0 -> Error number
                                <1>       ;     cf = 0 -> Cluster has been loaded at the buffer
                                <1>       ;
                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                                <1>
0000CE5A 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
                                <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
                                <1>
                                <1> read_file_sectors: ; 16/03/2016
0000CE5E 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
0000CE62 761C                   <1>       jna    short read_fs_cluster
                                <1>
                                <1> read_fat_file_sectors: ; 18/03/2016
0000CE64 83E802                 <1>       sub    eax, 2 ; Beginning cluster number is always 2
0000CE67 0FB65613               <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
0000CE6B F7E2                   <1>       mul    edx
0000CE6D 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>
0000CE70 E8085C0000             <1>       call   disk_read
0000CE75 7306                   <1>       jnc    short rclust_retn
                                <1>
                                <1>       ; 15/10/2016 (15h -> 17)
0000CE77 B811000000             <1>       mov    eax, 17 ; Drive not ready or read error !
0000CE7C C3                     <1>       retn
                                <1>
                                <1> rclust_retn:
0000CE7D 29C0                   <1>       sub    eax, eax ; 0
0000CE7F C3                     <1>       retn
                                <1>
                                <1> read_fs_cluster:
                                <1>       ; 15/02/2016 (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)
0000CE80 B980000000             <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
0000CE85 E801000000             <1>       call   read_fs_sectors
0000CE8A C3                     <1>       retn
                                <1>
                                <1> read_fs_sectors:
                                <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
0000CE8B F9                     <1>       stc
0000CE8C C3                     <1>       retn
                                <1>
                                <1> get_first_free_cluster:
                                <1>       ; 02/03/2016
                                <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 26/10/2010 (DRV_FAT.ASM, 'proc_get_first_free_cluster')
                                <1>       ; 10/07/2010
                                <1>       ; INPUT ->
                                <1>       ;     ESI = Logical DOS Drive Description Table address
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 1 -> Error code in AL (EAX)
                                <1>       ;     cf = 0 ->
                                <1>       ;       EAX = Cluster number
                                <1>       ;       If EAX = FFFFFFFFh -> no free space
                                <1>       ;     If the drive has FAT32 fs:
                                <1>       ;       EBX = FAT32 FSI sector buffer address (if > 0)
                                <1>
0000CE8D 8B4678                 <1>       mov    eax, [esi+LD_Clusters]
0000CE90 40                     <1>       inc    eax ; add eax, 1
0000CE91 A3[64930100]           <1>       mov    [gffc_last_free_cluster], eax
                                <1>
0000CE96 31DB                   <1>       xor    ebx, ebx ; 0 ; 02/03/2016
                                <1>
0000CE98 807E0302               <1>       cmp    byte [esi+LD_FATType], 2
0000CE9C 760E                   <1>       jna    short loc_gffc_get_first_fat_free_cluster0
                                <1>
                                <1> loc_gffc_get_first_fat32_free_cluster:
                                <1>       ; 02/03/2016
0000CE9E E844060000             <1>       call   get_fat32_fsinfo_sector_parms
0000CEA3 7207                   <1>       jc     short loc_gffc_get_first_fat_free_cluster0
                                <1>
                                <1> loc_gffc_check_fsinfo_parms:
                                <1>       ;;mov  ebx, DOSBootSectorBuff
                                <1>       ;cmp   dword [ebx], 41615252h
                                <1>       ;jne   short loc_gffc_fat32_fsinfo_err
                                <1>       ;cmp   dword [ebx+484], 61417272h
                                <1>       ;jne   short loc_gffc_fat32_fsinfo_err
                                <1>       ;mov   eax, [ebx+492] ; FSI_Next_Free
                                <1>       ;EAX = First free cluster
```

```
3549                                    <1>       ;(from FAT32 FSInfo sector)
3550 0000CEA5 89D0                      <1>       mov    eax, edx ; FSI_Next_Free (First Free Cluster)
3551 0000CEA7 83F8FF                    <1>       cmp    eax, 0FFFFFFFFh ; invalid (unknown) !
3552 0000CEAA 7205                      <1>       jb     short loc_gffc_get_first_fat_free_cluster1
3553                                    <1>
3554                                    <1>       ; Start from the 1st cluster of the FAT(32) file system
3555                                    <1> loc_gffc_get_first_fat_free_cluster0:
3556 0000CEAC B802000000                <1>       mov    eax, 2
3557                                    <1>       ;xor   edx, edx
3558                                    <1>
3559                                    <1> loc_gffc_get_first_fat_free_cluster1:
3560 0000CEB1 53                        <1>       push   ebx ; 02/03/2016
3561                                    <1>
3562                                    <1> loc_gffc_get_first_fat_free_cluster2:
3563 0000CEB2 A3[60930100]              <1>       mov    [gffc_first_free_cluster], eax
3564 0000CEB7 A3[5C930100]              <1>       mov    [gffc_next_free_cluster], eax
3565                                    <1>
3566                                    <1>       ; EBX = FAT32 FSINFO sector buffer address
3567                                    <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
3568                                    <1>       ; FAT32 FSINFO sector buffer is invalid.)
3569                                    <1>
3570                                    <1> loc_gffc_get_first_fat_free_cluster3:
3571 0000CEBC E875FDFFFF                <1>       call   get_next_cluster
3572 0000CEC1 7307                      <1>       jnc    short loc_gffc_get_first_fat_free_cluster4
3573 0000CEC3 09C0                      <1>       or     eax, eax
3574 0000CEC5 740B                      <1>       jz     short loc_gffc_first_free_fat_cluster_next
3575 0000CEC7 5B                        <1>       pop    ebx ; 02/03/2016
3576 0000CEC8 F5                        <1>       cmc    ; stc
3577 0000CEC9 C3                        <1>       retn
3578                                    <1>
3579                                    <1> loc_gffc_get_first_fat_free_cluster4:
3580 0000CECA 21C0                      <1>       and    eax, eax ; next cluster value
3581 0000CECC 7504                      <1>       jnz    short loc_gffc_first_free_fat_cluster_next
3582 0000CECE 89C8                      <1>       mov    eax, ecx ; current (previous cluster) value
3583 0000CED0 EB22                      <1>       jmp    short loc_gffc_check_for_set
3584                                    <1>
3585                                    <1> loc_gffc_first_free_fat_cluster_next:
3586 0000CED2 A1[5C930100]              <1>       mov    eax, [gffc_next_free_cluster]
3587 0000CED7 3B05[64930100]            <1>       cmp    eax, [gffc_last_free_cluster]
3588 0000CEDD 7308                      <1>       jnb    short retn_stc_from_get_first_free_cluster
3589                                    <1> pass_gffc_last_cluster_eax_check:
3590 0000CEDF 40                        <1>       inc    eax ; add eax, 1
3591 0000CEE0 A3[5C930100]              <1>       mov    [gffc_next_free_cluster], eax
3592 0000CEE5 EBD5                      <1>       jmp    short loc_gffc_get_first_fat_free_cluster3
3593                                    <1>
3594                                    <1> retn_stc_from_get_first_free_cluster:
3595 0000CEE7 A1[60930100]              <1>       mov    eax, [gffc_first_free_cluster]
3596 0000CEEC 83F802                    <1>       cmp    eax, 2
3597 0000CEEF 7709                      <1>       ja     short loc_gffc_check_previous_clusters
3598 0000CEF1 29C0                      <1>       sub    eax, eax
3599 0000CEF3 48                        <1>       dec    eax ; FFFFFFFFh
3600                                    <1>
3601                                    <1> loc_gffc_check_for_set:
3602                                    <1>       ; 02/03/2016
3603 0000CEF4 5B                        <1>       pop    ebx
3604                                    <1>
3605                                    <1>       ; EBX = FAT32 FSINFO sector buffer address
3606                                    <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
3607                                    <1>       ; FAT32 FSINFO sector buffer is invalid.)
3608                                    <1>
3609 0000CEF5 09DB                      <1>       or     ebx, ebx
3610 0000CEF7 750E                      <1>       jnz    short loc_gffc_set_ffree_fat32_cluster
3611                                    <1>
3612                                    <1>       ;cmp   byte [esi+LD_FATType], 3
3613                                    <1>       ;jnb   short loc_gffc_set_ffree_fat32_cluster
3614                                    <1>
3615                                    <1>       ;xor   ebx, ebx ; 0
3616                                    <1>
3617                                    <1> loc_gffc_retn:
3618 0000CEF9 C3                        <1>       retn
3619                                    <1>
3620                                    <1> loc_gffc_check_previous_clusters:
3621 0000CEFA 48                        <1>       dec    eax ; sub eax, 1
3622 0000CEFB A3[64930100]              <1>       mov    [gffc_last_free_cluster], eax
3623 0000CF00 B802000000                <1>       mov    eax, 2
3624                                    <1>       ;xor   edx, edx
3625 0000CF05 EBAB                      <1>       jmp    short loc_gffc_get_first_fat_free_cluster2
3626                                    <1>
3627                                    <1> loc_gffc_set_ffree_fat32_cluster:
3628                                    <1>       ;call  set_first_free_cluster
3629                                    <1>       ;retn
3630                                    <1>       ;jmp   short set_first_free_cluster
3631                                    <1>
3632                                    <1> set_first_free_cluster:
3633                                    <1>       ; 15/10/2016
3634                                    <1>       ; 23/03/2016
3635                                    <1>       ; 02/03/2016
3636                                    <1>       ; 29/02/2016
3637                                    <1>       ; 26/02/2016
3638                                    <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
3639                                    <1>       ; 21/08/2011 (DRV_FAT.ASM, 'proc_set_first_free_cluster')
3640                                    <1>       ; 11/07/2010
3641                                    <1>       ; INPUT ->
3642                                    <1>       ;      ESI = Logical DOS Drive Description Table address
3643                                    <1>       ;      EAX = First free cluster
3644                                    <1>       ;      EBX = FSINFO sector buffer address
3645                                    <1>       ;      ;;If EBX > 0, it is FSINFO sector buffer address
3646                                    <1>       ;      ;;EBX = 0, if FSINFO sector is not loaded
3647                                    <1>       ; OUTPUT->
3648                                    <1>       ;      ESI = Logical DOS Drive Description Table address
3649                                    <1>       ;      If EBX > 0, it is FSINFO sector buffer address
3650                                    <1>       ;      EBX = 0, if FSINFO sector could not be loaded
3651                                    <1>       ;      CF = 1 -> Error code in AL (EAX)
3652                                    <1>       ;      CF = 0 -> first free cluster is successfully updated
3653                                    <1>
```

```
3654                                    <1>       ;cmp   byte [esi+LD_FATType], 3
3655                                    <1>       ;jb    short loc_sffc_invalid_drive
3656                                    <1>
3657                                    <1>       ; Save First Free Cluster value for 'update_cluster'
3658 0000CF07 89463E                   <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First free Cluster
3659                                    <1>
3660                                    <1>       ;or    ebx, ebx
3661                                    <1>       ;jnz   short loc_sffc_read_fsinfo_sector
3662                                    <1>
3663 0000CF0A 813B52526141             <1>       cmp     dword [ebx], 41615252h
3664 0000CF10 7540                     <1>       jne    short loc_sffc_read_fsinfo_sector
3665 0000CF12 81BBE4010000727241-      <1>       cmp    dword [ebx+484], 61417272h
3665 0000CF1B 61                       <1>
3666 0000CF1C 7534                     <1>       jne    short loc_sffc_read_fsinfo_sector
3667                                    <1>
3668 0000CF1E 3B83EC010000             <1>       cmp    eax, [ebx+492]  ; FSI_Next_Free
3669 0000CF24 741F                     <1>       je     short loc_sffc_retn
3670                                    <1>
3671                                    <1> loc_sffc_write_fsinfo_sector:
3672                                    <1>       ; EBX = FSINFO sector buffer
3673                                    <1>       ; [CFS_FAT32FSINFOSEC] is set in 'get_fat32_fsinfo_sector_parms'
3674 0000CF26 8983EC010000             <1>       mov    [ebx+492], eax
3675 0000CF2C A1[74930100]             <1>       mov    eax, [CFS_FAT32FSINFOSEC]
3676 0000CF31 B901000000               <1>       mov    ecx, 1
3677 0000CF36 53                       <1>       push   ebx
3678 0000CF37 E8325B0000               <1>       call   disk_write
3679 0000CF3C 7208                     <1>       jc      short loc_sffc_read_fsinfo_sector_err1
3680 0000CF3E 5B                       <1>       pop    ebx
3681                                    <1>
3682 0000CF3F 8B83EC010000             <1>       mov    eax, [ebx+492] ; First (Next) Free Cluster
3683                                    <1>
3684                                    <1> loc_sffc_retn:
3685 0000CF45 C3                       <1>       retn
3686                                    <1>
3687                                    <1> ;loc_sffc_invalid_drive:
3688                                    <1> ;     mov    eax, 0Fh ; MSDOS Error : Invalid drive
3689                                    <1> ;     push   edx
3690                                    <1>
3691                                    <1> loc_sffc_read_fsinfo_sector_err1:
3692 0000CF46 BB00000000               <1>       mov    ebx, 0
3693                                    <1>       ; 15/10/2016 (1Dh -> 18)
3694                                    <1>       ; 23/03/2016 (1Dh)
3695 0000CF4B B812000000               <1>       mov    eax, 18 ; Drive not ready or write error
3696                                    <1>
3697                                    <1> loc_sffc_read_fsinfo_sector_err2:
3698 0000CF50 5A                       <1>       pop    edx
3699 0000CF51 C3                       <1>       retn
3700                                    <1>
3701                                    <1> loc_sffc_read_fsinfo_sector:
3702 0000CF52 50                       <1>       push   eax
3703                                    <1>
3704 0000CF53 E88F050000               <1>       call   get_fat32_fsinfo_sector_parms
3705 0000CF58 72F6                     <1>       jc     short loc_sffc_read_fsinfo_sector_err2
3706                                    <1>
3707 0000CF5A 58                       <1>       pop    eax
3708                                    <1>       ; EDX = First (Next) Free Cluster value from FSINFO sector
3709                                    <1>       ; EAX = First Free Cluster value from 'get_next_cluster'
3710                                    <1>       ; (edx = old value)
3711 0000CF5B 39D0                     <1>       cmp    eax, edx ; First free Cluster (eax = new value)
3712 0000CF5D 75C7                     <1>       jne    short loc_sffc_write_fsinfo_sector
3713                                    <1>
3714 0000CF5F C3                       <1>       retn
3715                                    <1>
3716                                    <1> update_cluster:
3717                                    <1>       ; 23/10/2016
3718                                    <1>       ; 23/03/2016
3719                                    <1>       ; 02/03/2016
3720                                    <1>       ; 01/03/2016
3721                                    <1>       ; 29/02/2016
3722                                    <1>       ; 27/02/2016
3723                                    <1>       ; 26/02/2016
3724                                    <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
3725                                    <1>       ; 11/08/2011
3726                                    <1>       ; 09/02/2005
3727                                    <1>       ; INPUT ->
3728                                    <1>       ;      EAX = Cluster Number
3729                                    <1>       ;      ECX = New Cluster Value
3730                                    <1>       ;      ESI = Logical Dos Drive Parameters Table
3731                                    <1>       ;
3732                                    <1>       ;      /// dword [FAT_ClusterCounter] ///
3733                                    <1>       ;
3734                                    <1>       ; OUTPUT ->
3735                                    <1>       ;      cf = 0 -> No Error, EAX is valid
3736                                    <1>       ;      cf = 1 & EAX = 0 -> End Of Cluster Chain
3737                                    <1>       ;      cf = 1 & EAX > 0 -> Error
3738                                    <1>       ;            (ECX -> any value)
3739                                    <1>       ;      EAX = Next Cluster
3740                                    <1>       ;      ECX = New Cluster Value
3741                                    <1>       ;
3742                                    <1>       ;      /// [FAT_ClusterCounter] is updated,
3743                                    <1>       ;      /// decreased when a free cluster is assigned,
3744                                    <1>       ;      /// increased if an assigned cluster is freed.
3745                                    <1>       ;
3746                                    <1>       ;
3747                                    <1>       ; (Modified registers: EAX, EBX, -ECX-, EDX)
3748                                    <1>
3749 0000CF60 A3[C6900100]             <1>       mov    [FAT_CurrentCluster], eax
3750 0000CF65 890D[68930100]           <1>       mov    [ClusterValue], ecx
3751                                    <1>
3752                                    <1> loc_update_cluster_check_fat_buffer:
3753 0000CF6B 8A1E                     <1>       mov    bl, [esi+LD_Name]
3754 0000CF6D 381D[CB900100]           <1>       cmp    [FAT_BuffDrvName], bl
3755 0000CF73 741A                     <1>       je     short loc_update_cluster_check_fat_type
3756 0000CF75 803D[CA900100]02         <1>       cmp    byte [FAT_BuffValidData], 2
3757 0000CF7C 0F84C2000000             <1>         je      loc_uc_save_fat_buffer
```

```
                                <1>
                                <1> loc_uc_reset_fat_buffer_validation:
0000CF82 C605[CA900100]00       <1>       mov   byte [FAT_BuffValidData], 0
                                <1>
                                <1> loc_uc_check_fat_type_reset_drvname:
0000CF89 881D[CB900100]         <1>       mov   [FAT_BuffDrvName], bl
                                <1>
                                <1> loc_update_cluster_check_fat_type:
0000CF8F 29D2                   <1>       sub   edx, edx ; 26/02/2016
0000CF91 8A5E03                 <1>       mov   bl, [esi+LD_FATType]
0000CF94 83F802                 <1>       cmp   eax, 2
0000CF97 0F82BE000000           <1>         jb      update_cluster_inv_data
0000CF9D 80FB02                 <1>       cmp   bl, 2
0000CFA0 0F877A010000           <1>         ja      update_fat32_cluster
                                <1>       ;cmp  bl, 1
                                <1>       ;jb   short update_cluster_inv_data
0000CFA6 8B4E78                 <1>       mov   ecx, [esi+LD_Clusters]
0000CFA9 41                     <1>       inc   ecx
0000CFAA 890D[D6900100]         <1>       mov   [LastCluster], ecx
0000CFB0 39C8                   <1>       cmp   eax, ecx ; dword [LastCluster]
0000CFB2 0F87A6000000           <1>         ja      return_uc_fat_stc
                                <1>       ; TRDOS v1 has a FATal bug here !
                                <1>             ; or bl, bl ; cmp bl, 0
                                <1>             ; jz short update_fat12_cluster
                                <1>       ; !! It would destroy FAT12 floppy disk fs here !!
                                <1>       ; ('A:' disks of TRDOS v1 operating system project
                                <1>       ; had 'singlix fs', so, I could not differ this mistake
                                <1>       ; on a drive 'A:')
0000CFB8 80FB01                 <1>       cmp   bl, 1 ; correct comparison is this !
0000CFBB 0F86A2000000           <1>         jna     update_fat12_cluster
                                <1>
                                <1> update_fat16_cluster:
                                <1> pass_uc_fat16_errc:
                                <1>       ;sub   edx, edx
0000CFC1 BB00030000             <1>       mov   ebx, 300h ;768
0000CFC6 F7F3                   <1>       div   ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; DX = Cluster offset in FAT buffer
0000CFC8 6689D3                 <1>       mov   bx, dx
0000CFCB 66D1E3                 <1>       shl   bx, 1 ; Multiply by 2
0000CFCE 66BA0300               <1>       mov   dx, 3
0000CFD2 F7E2                   <1>       mul   edx
                                <1>       ; EAX = FAT Sector
                                <1>       ; EDX = 0
                                <1>       ; EBX = Byte offset in FAT buffer
0000CFD4 8A0D[CA900100]         <1>       mov   cl, [FAT_BuffValidData]
0000CFDA 80F902                 <1>       cmp   cl, 2
0000CFDD 750A                   <1>       jne   short loc_uc_check_fat16_buff_sector_load
                                <1>
                                <1> loc_uc_check_fat16_buff_sector_save:
0000CFDF 3B05[CE900100]         <1>       cmp   eax, [FAT_BuffSector]
0000CFE5 755D                   <1>       jne   short loc_uc_save_fat_buffer
0000CFE7 EB15                   <1>       jmp   short loc_update_fat16_cell
                                <1>
                                <1> loc_uc_check_fat16_buff_sector_load:
0000CFE9 80F901                 <1>       cmp   cl, 1 ; byte [FAT_BuffValidData]
0000CFEC 0F85FB010000           <1>         jne     loc_uc_load_fat_sectors
0000CFF2 3B05[CE900100]         <1>       cmp   eax, [FAT_BuffSector]
0000CFF8 0F85EF010000           <1>         jne     loc_uc_load_fat_sectors
                                <1>
                                <1> loc_update_fat16_cell:
                                <1> loc_update_fat16_buffer:
0000CFFE 81C3001C0900           <1>       add   ebx, FAT_Buffer ; 26/02/2016
                                <1>       ;movzx eax, word [ebx]
0000D004 668B03                 <1>       mov   ax, [ebx]
                                <1>       ; 01/03/2016
0000D007 89C2                   <1>       mov   edx, eax ; old value of the cluster
0000D009 A3[C6900100]           <1>       mov   [FAT_CurrentCluster], eax
0000D00E 8B0D[68930100]         <1>       mov   ecx, [ClusterValue] ; 32 bits
0000D014 66890B                 <1>       mov   [ebx], cx ; 16 bits !
                                <1>
0000D017 C605[CA900100]02       <1>       mov   byte [FAT_BuffValidData], 2
                                <1>
0000D01E 6683F802               <1>       cmp   ax, 2
0000D022 723A                   <1>       jb    short return_uc_fat_stc
0000D024 3B05[D6900100]         <1>       cmp   eax, [LastCluster]
0000D02A 7732                   <1>       ja    short return_uc_fat_stc
                                <1>
                                <1> loc_fat_buffer_updated:
                                <1>       ; 01/03/2016
0000D02C F8                     <1>       clc
                                <1> loc_fat_buffer_stc_1:
0000D02D 9C                     <1>       pushf
0000D02E 21C9                   <1>       and   ecx, ecx
0000D030 7506                   <1>       jnz   short loc_fat_buffer_updated_1
                                <1>
                                <1>       ; 01/03/2016
                                <1>       ; new value of the cluster = 0 (free)
                                <1>       ; increase free(d) cluster count
0000D032 FF05[D2900100]         <1>       inc   dword [FAT_ClusterCounter]
                                <1>
                                <1> loc_fat_buffer_updated_1: ; new value of the cluster > 0
0000D038 09D2                   <1>       or    edx, edx ; 02/03/2016
0000D03A 7506                   <1>       jnz   short loc_fat_buffer_updated_2
                                <1>       ; old value of the cluster = 0 (it was free cluster)
                                <1>       ; decrease free(d) cluster count
0000D03C FF0D[D2900100]         <1>       dec   dword [FAT_ClusterCounter] ; it may be negative number
                                <1>
                                <1> loc_fat_buffer_updated_2:
0000D042 9D                     <1>       popf
0000D043 C3                     <1>       retn
                                <1>
                                <1> loc_uc_save_fat_buffer:
                                <1>       ; byte [FAT_BuffValidData] = 2
0000D044 E8D4010000             <1>       call  save_fat_buffer
```

```
3863 0000D049 0F8297010000        <1>         jc      loc_fat_sectors_rw_error2
3864                              <1>       ;mov   byte [FAT_BuffValidData], 1
3865 0000D04F A1[C6900100]        <1>       mov    eax, [FAT_CurrentCluster]
3866                              <1>       ;mov   ecx, [ClusterValue]
3867                              <1>       ;jmp   short loc_update_cluster_check_fat_buffer
3868 0000D054 8A1E                <1>       mov    bl, [esi+LD_Name] ; 01/03/2016
3869 0000D056 E927FFFFFF          <1>         jmp     loc_uc_reset_fat_buffer_validation
3870                              <1>
3871                              <1> update_cluster_inv_data:
3872                              <1>       ;mov   eax, 0Dh
3873 0000D05B B00D                <1>       mov    al, 0Dh  ; Invalid Data
3874 0000D05D C3                  <1>       retn
3875                              <1>
3876                              <1> return_uc_fat_stc:
3877                              <1>       ; 01/03/2016
3878 0000D05E 31C0                <1>       xor    eax, eax
3879 0000D060 F9                  <1>       stc
3880 0000D061 EBCA                <1>       jmp    short loc_fat_buffer_stc_1
3881                              <1>
3882                              <1> update_fat12_cluster:
3883                              <1> pass_uc_fat12_errc:
3884                              <1>       ;sub   edx, edx
3885 0000D063 BB00040000          <1>       mov    ebx, 400h ;1024
3886 0000D068 F7F3                <1>       div    ebx
3887                              <1>       ; EAX = Count of 3 FAT sectors
3888                              <1>       ; DX = Cluster offset in FAT buffer
3889 0000D06A 66B90300            <1>       mov    cx, 3
3890 0000D06E 6689C3              <1>       mov    bx, ax
3891 0000D071 6689C8              <1>       mov    ax, cx ; 3
3892 0000D074 66F7E2              <1>       mul    dx     ; Multiply by 3
3893 0000D077 66D1E8              <1>       shr    ax, 1  ; Divide by 2
3894 0000D07A 6693                <1>       xchg   bx, ax
3895                              <1>       ; EAX = Count of 3 FAT sectors
3896                              <1>       ; EBX = Byte Offset in FAT buffer
3897 0000D07C 66F7E1              <1>       mul    cx  ; 3 * AX
3898                              <1>       ; EAX = FAT Beginning Sector
3899                              <1>       ; EDX = 0
3900 0000D07F 8A0D[CA900100]      <1>       mov    cl, [FAT_BuffValidData]
3901                              <1>       ; TRDOS v1 has a FATal bug here !
3902                              <1>       ; (it does not have 'cmp cl, 2' instruction here !
3903                              <1>       ;  while 'jne' is existing !)
3904 0000D085 80F902              <1>       cmp    cl, 2 ; 2 = dirty buffer (must be written to disk)
3905 0000D088 750A                <1>       jne    short loc_uc_check_fat12_buff_sector_load
3906                              <1>
3907                              <1> loc_uc_check_fat12_buff_sector_save:
3908 0000D08A 3B05[CE900100]      <1>       cmp    eax, [FAT_BuffSector]
3909 0000D090 75B2                <1>         jne     short loc_uc_save_fat_buffer
3910 0000D092 EB15                <1>       jmp    short loc_update_fat12_cell
3911                              <1>
3912                              <1> loc_uc_check_fat12_buff_sector_load:
3913 0000D094 80F901              <1>       cmp    cl, 1 ; byte ptr [FAT_BuffValidData]
3914 0000D097 0F8550010000        <1>         jne     loc_uc_load_fat_sectors
3915 0000D09D 3B05[CE900100]      <1>       cmp    eax, [FAT_BuffSector]
3916 0000D0A3 0F8544010000        <1>         jne     loc_uc_load_fat_sectors
3917                              <1>
3918                              <1> loc_update_fat12_cell:
3919 0000D0A9 81C3001C0900        <1>       add    ebx, FAT_Buffer ; 26/02/2016
3920 0000D0AF 668B0D[C6900100]    <1>       mov    cx, [FAT_CurrentCluster]
3921 0000D0B6 66D1E9              <1>       shr    cx, 1
3922 0000D0B9 668B03              <1>       mov    ax, [ebx]
3923 0000D0BC 6689C2              <1>       mov    dx, ax
3924 0000D0BF 7344                <1>       jnc    short uc_fat12_nc_even
3925                              <1>
3926 0000D0C1 6683E00F            <1>       and    ax, 0Fh
3927 0000D0C5 8B0D[68930100]      <1>       mov    ecx, [ClusterValue] ; 32 bits
3928 0000D0CB 66C1E104            <1>       shl    cx, 4
3929 0000D0CF 6609C1              <1>       or     cx, ax
3930 0000D0D2 6689D0              <1>       mov    ax, dx
3931 0000D0D5 66890B              <1>       mov    [ebx], cx  ; 16 bits !
3932 0000D0D8 66C1E804            <1>       shr    ax, 4 ; al(bit4..7)+ah(bit0..7)
3933                              <1>
3934                              <1> update_fat12_buffer:
3935 0000D0DC A3[C6900100]        <1>       mov    [FAT_CurrentCluster], eax
3936 0000D0E1 89C2                <1>       mov    edx, eax ; 01/03/2016
3937 0000D0E3 C605[CA900100]02    <1>       mov    byte [FAT_BuffValidData], 2
3938 0000D0EA 6683F802            <1>       cmp    ax, 2
3939 0000D0EE 0F826AFFFFFF        <1>         jb      return_uc_fat_stc
3940 0000D0F4 3B05[D6900100]      <1>       cmp    eax, [LastCluster]
3941 0000D0FA 0F875EFFFFFF        <1>         ja      return_uc_fat_stc
3942 0000D100 E927FFFFFF          <1>         jmp     loc_fat_buffer_updated
3943                              <1>
3944                              <1> uc_fat12_nc_even:
3945 0000D105 662500F0            <1>       and    ax, 0F000h
3946 0000D109 8B0D[68930100]      <1>       mov    ecx, [ClusterValue] ; 32 bits
3947 0000D10F 80E50F              <1>       and    ch, 0Fh
3948 0000D112 6609C1              <1>       or     cx, ax
3949 0000D115 6689D0              <1>       mov    ax, dx
3950 0000D118 66890B              <1>       mov    [ebx], cx ; 16 bits !
3951 0000D11B 80E40F              <1>       and    ah, 0Fh ; al(bit0..7)+ah(bit0..3)
3952 0000D11E EBBC                <1>       jmp    short update_fat12_buffer
3953                              <1>
3954                              <1> update_fat32_cluster:
3955 0000D120 8B4E78              <1>       mov    ecx, [esi+LD_Clusters]
3956 0000D123 41                  <1>       inc    ecx
3957 0000D124 890D[D6900100]      <1>       mov    [LastCluster], ecx
3958                              <1>
3959 0000D12A 39C8                <1>       cmp    eax, ecx
3960 0000D12C 0F872CFFFFFF        <1>         ja      return_uc_fat_stc
3961                              <1>
3962                              <1> pass_uc_fat32_errc:
3963                              <1>       ;sub   edx, edx
3964 0000D132 BB80010000          <1>       mov    ebx, 180h ;384
3965 0000D137 F7F3                <1>       div    ebx
3966                              <1>       ; EAX = Count of 3 FAT sectors
3967                              <1>       ; DX = Cluster offset in FAT buffer
```

```
3968 0000D139 89D3              <1>       mov    ebx, edx
3969 0000D13B C1E302            <1>       shl    ebx, 2 ; Multiply by 4
3970 0000D13E BA03000000        <1>       mov    edx, 3
3971 0000D143 F7E2              <1>       mul    edx
3972                            <1>       ; EBX = Cluster Offset in FAT buffer
3973                            <1>       ; EAX = FAT Sector
3974                            <1>       ; EDX = 0
3975 0000D145 8A0D[CA900100]    <1>       mov    cl, [FAT_BuffValidData]
3976 0000D14B 80F902            <1>       cmp    cl, 2
3977 0000D14E 750E              <1>       jne    short loc_uc_check_fat32_buff_sector_load
3978                            <1>
3979                            <1> loc_uc_check_fat32_buff_sector_save:
3980 0000D150 3B05[CE900100]    <1>       cmp    eax, [FAT_BuffSector]
3981 0000D156 0F85E8FEFFFF      <1>         jne     loc_uc_save_fat_buffer
3982 0000D15C EB11              <1>       jmp    short loc_update_fat32_cell
3983                            <1>
3984                            <1> loc_uc_check_fat32_buff_sector_load:
3985 0000D15E 80F901            <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
3986 0000D161 0F8586000000      <1>         jne     loc_uc_load_fat_sectors
3987 0000D167 3B05[CE900100]    <1>       cmp    eax, [FAT_BuffSector]
3988 0000D16D 757E              <1>         jne     loc_uc_load_fat_sectors
3989                            <1>
3990                            <1> loc_update_fat32_cell:
3991                            <1> loc_update_fat32_buffer:
3992 0000D16F 81C3001C0900      <1>       add    ebx, FAT_Buffer ; 26/02/2016
3993 0000D175 8B03              <1>       mov    eax, [ebx]
3994 0000D177 25FFFFFF0F        <1>       and    eax, 0FFFFFFFh ; 28 bit cluster value
3995                            <1>
3996 0000D17C 8B15[C6900100]    <1>       mov    edx, [FAT_CurrentCluster] ; 01/03/2016
3997                            <1>
3998 0000D182 A3[C6900100]      <1>       mov    [FAT_CurrentCluster], eax
3999 0000D187 8B0D[68930100]    <1>       mov    ecx, [ClusterValue]
4000 0000D18D 890B              <1>       mov    [ebx], ecx ; 29/02/2016
4001                            <1>
4002 0000D18F C605[CA900100]02  <1>       mov    byte [FAT_BuffValidData], 2
4003                            <1>
4004                            <1>       ; 01/03/2016
4005 0000D196 21C0              <1>       and    eax, eax ; was it free cluster ?
4006 0000D198 7514              <1>       jnz    short loc_upd_fat32_c0
4007                            <1>
4008                            <1>       ;or    ecx, ecx ; it will be left free ?!
4009                            <1>       ;jz    short loc_upd_fat32_c3
4010                            <1>
4011 0000D19A 3B563E            <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
4012 0000D19D 7520              <1>       jne    short loc_upd_fat32_c3
4013                            <1>
4014 0000D19F 3B15[D6900100]    <1>       cmp    edx, [LastCluster]
4015 0000D1A5 7207              <1>       jb     short loc_upd_fat32_c0
4016                            <1>
4017 0000D1A7 BA02000000        <1>       mov    edx, 2 ; rewind !
4018 0000D1AC EB0E              <1>       jmp    short loc_upd_fat32_c2
4019                            <1>
4020                            <1> loc_upd_fat32_c0:
4021 0000D1AE FF463E            <1>       inc    dword [esi+LD_BPB+BPB_Reserved+4] ; set it to next cluster
4022 0000D1B1 EB0C              <1>       jmp    short loc_upd_fat32_c3
4023                            <1>
4024                            <1> loc_upd_fat32_c1:
4025 0000D1B3 09C9              <1>       or     ecx, ecx ; will it be free cluster ?
4026 0000D1B5 7508              <1>       jnz    short loc_upd_fat32_c3
4027                            <1>
4028 0000D1B7 3B563E            <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
4029 0000D1BA 7303              <1>       jnb    short loc_upd_fat32_c3
4030                            <1>
4031                            <1> loc_upd_fat32_c2:
4032 0000D1BC 89563E            <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
4033                            <1>
4034                            <1> loc_upd_fat32_c3:
4035 0000D1BF 89C2              <1>       mov    edx, eax
4036                            <1>
4037                            <1> loc_upd_fat32_c4:
4038 0000D1C1 83F802            <1>       cmp    eax, 2
4039 0000D1C4 0F8294FEFFFF      <1>         jb      return_uc_fat_stc
4040                            <1>
4041                            <1> pass_uc_fat32_c_zero_check_2:
4042 0000D1CA 3B05[D6900100]    <1>       cmp    eax, [LastCluster]
4043 0000D1D0 0F8788FEFFFF      <1>         ja      return_uc_fat_stc
4044                            <1>
4045 0000D1D6 E951FEFFFF        <1>       jmp     loc_fat_buffer_updated
4046                            <1>
4047                            <1> loc_fat_sectors_rw_error1:
4048                            <1>       ;mov   byte [FAT_BuffValidData], 0
4049                            <1>       ; 23/10/2016 (15h -> 17)
4050                            <1>       ; 23/03/2016
4051 0000D1DB B811000000        <1>       mov    eax, 17 ; Drive not ready or read error
4052 0000D1E0 8825[CA900100]    <1>       mov    [FAT_BuffValidData], ah ; 0
4053                            <1>
4054                            <1> loc_fat_sectors_rw_error2:
4055                            <1>       ;mov   eax, error code
4056                            <1>       ;mov   edx, 0
4057 0000D1E6 8B0D[68930100]    <1>       mov    ecx, [ClusterValue]
4058 0000D1EC C3                <1>       retn
4059                            <1>
4060                            <1> loc_uc_load_fat_sectors:
4061 0000D1ED A3[CE900100]      <1>       mov    [FAT_BuffSector], eax
4062                            <1>
4063                            <1> load_uc_fat_sectors_zero:
4064 0000D1F2 034660            <1>       add    eax, [esi+LD_FATBegin]
4065 0000D1F5 BB001C0900        <1>       mov    ebx, FAT_Buffer
4066 0000D1FA B903000000        <1>       mov    ecx, 3
4067 0000D1FF E879580000        <1>       call   disk_read
4068 0000D204 72D5              <1>       jc     short loc_fat_sectors_rw_error1
4069                            <1>
4070 0000D206 C605[CA900100]01  <1>         mov     byte [FAT_BuffValidData], 1
4071 0000D20D A1[C6900100]      <1>       mov    eax, [FAT_CurrentCluster]
4072 0000D212 8B0D[68930100]    <1>       mov    ecx, [ClusterValue]
```

```
4073 0000D218 E972FDFFFF          <1>         jmp     loc_update_cluster_check_fat_type
4074                              <1>
4075                              <1> save_fat_buffer:
4076                              <1>         ; 15/10/2016
4077                              <1>         ; 01/03/2016
4078                              <1>         ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
4079                              <1>         ; 11/08/2011
4080                              <1>         ; 09/02/2005
4081                              <1>         ; INPUT ->
4082                              <1>         ;      None
4083                              <1>         ; OUTPUT ->
4084                              <1>         ;      cf = 0 -> OK.
4085                              <1>         ;      cf = 1 -> error code in AL (EAX)
4086                              <1>         ;
4087                              <1>         ;      EBX = FAT_Buffer address
4088                              <1>         ;
4089                              <1>         ; (EAX, EDX, ECX will be modified)
4090                              <1>
4091                              <1>         ;cmp   byte [FAT_BuffValidData], 2
4092                              <1>         ;je    short loc_save_fat_buff
4093                              <1>
4094                              <1> ;loc_save_fat_buffer_retn:
4095                              <1> ;     xor    eax, eax
4096                              <1> ;     retn
4097                              <1>
4098                              <1> loc_save_fat_buff:
4099 0000D21D 31D2                <1>         xor    edx, edx
4100 0000D21F 8A35[CB900100]      <1>         mov    dh, [FAT_BuffDrvName]
4101 0000D225 80FE41              <1>         cmp    dh, 'A'
4102 0000D228 722E                <1>         jb     short loc_save_fat_buffer_inv_data_retn
4103 0000D22A 80EE41              <1>         sub    dh, 'A'
4104 0000D22D 56                  <1>         push   esi ; *
4105 0000D22E BE00010900          <1>           mov     esi, Logical_DOSDisks
4106 0000D233 01D6                <1>         add    esi, edx
4107                              <1>
4108 0000D235 8A5603              <1>         mov    dl, [esi+LD_FATType]
4109 0000D238 20D2                <1>         and    dl, dl
4110 0000D23A 741B                <1>         jz     short loc_save_fat_buffer_inv_data_pop_retn
4111                              <1>
4112 0000D23C A1[CE900100]        <1>         mov    eax, [FAT_BuffSector]
4113 0000D241 80FA02              <1>         cmp    dl, 2
4114 0000D244 770A                <1>         ja     short loc_save_fat32_buff
4115                              <1>
4116                              <1> loc_save_fat_12_16_buff:
4117                              <1>         ; 01/03/2016
4118                              <1>         ; TRDOS v1 has a FATal bug here!
4119                              <1>         ; Correct code: mov dx, word ptr [FAT_BuffSector]+2
4120                              <1>         ; (DX:AX in TRDOS v1 -> EAX in TRDOS v2)
4121                              <1>         ;
4122 0000D246 0FB74E1C            <1>         movzx  ecx, word [esi+LD_BPB+FATSecs]
4123 0000D24A 29C1                <1>         sub    ecx, eax
4124                              <1>         ; TRDOS v1 has a bug here... ('pop esi' was forgotten!)
4125                              <1>         ;jna   short loc_save_fat_buffer_inv_data_retn ; wrong addr!
4126 0000D24C 7609                <1>         jna    short loc_save_fat_buffer_inv_data_pop_retn ; correct addr.
4127 0000D24E EB15                <1>         jmp    short loc_save_fat_buffer_check_rs3
4128                              <1>
4129                              <1> loc_save_fat32_buff:
4130 0000D250 8B4E2A              <1>         mov    ecx, [esi+LD_BPB+FAT32_FAT_Size]
4131 0000D253 29C1                <1>         sub    ecx, eax
4132 0000D255 770E                <1>         ja     short loc_save_fat_buffer_check_rs3
4133                              <1>
4134                              <1> loc_save_fat_buffer_inv_data_pop_retn:
4135 0000D257 5E                  <1>         pop    esi ; *
4136                              <1> loc_save_fat_buffer_inv_data_retn:
4137 0000D258 B80D000000          <1>         mov    eax, 0Dh ; Invalid DATA
4138 0000D25D C3                  <1>         retn
4139                              <1>
4140                              <1> loc_save_fat_buff_remain_sectors_3:
4141 0000D25E B903000000          <1>         mov    ecx, 3
4142 0000D263 EB05                <1>         jmp    short loc_save_fat_buff_continue
4143                              <1>
4144                              <1> loc_save_fat_buffer_check_rs3:
4145 0000D265 83F903              <1>         cmp    ecx, 3
4146 0000D268 77F4                <1>         ja     short loc_save_fat_buff_remain_sectors_3
4147                              <1>
4148                              <1> loc_save_fat_buff_continue:
4149 0000D26A BB001C0900          <1>         mov    ebx, FAT_Buffer
4150 0000D26F 034660              <1>         add    eax, [esi+LD_FATBegin]
4151 0000D272 51                  <1>         push   ecx
4152 0000D273 E8F6570000          <1>         call   disk_write
4153 0000D278 59                  <1>         pop    ecx
4154 0000D279 722B                <1>         jc     short loc_save_FAT_buff_write_err
4155                              <1>
4156 0000D27B 807E0302            <1>         cmp    byte [esi+LD_FATType], 2
4157 0000D27F 7605                <1>         jna    short loc_calc_2nd_fat12_16_addr
4158                              <1>
4159                              <1> loc_calc_2nd_fat32_addr:
4160 0000D281 8B462A              <1>         mov    eax, [esi+LD_BPB+FAT32_FAT_Size]
4161 0000D284 EB04                <1>         jmp    short loc_calc_2nd_fat_addr
4162                              <1>
4163                              <1> loc_calc_2nd_fat12_16_addr:
4164 0000D286 0FB7461C            <1>         movzx  eax, word [esi+LD_BPB+FATSecs]
4165                              <1>
4166                              <1> loc_calc_2nd_fat_addr:
4167 0000D28A 034660              <1>         add    eax, [esi+LD_FATBegin]
4168 0000D28D 0305[CE900100]      <1>         add    eax, [FAT_BuffSector]
4169 0000D293 BB001C0900          <1>         mov    ebx, FAT_Buffer
4170                              <1>         ; ecx = 1 to 3
4171 0000D298 E8D1570000          <1>         call   disk_write
4172 0000D29D 7207                <1>         jc     short loc_save_FAT_buff_write_err
4173                              <1>         ; Valid  buffer (1 = valid but do not save)
4174 0000D29F C605[CA900100]01    <1>         mov    byte [FAT_BuffValidData], 1
4175                              <1>
4176                              <1> loc_save_FAT_buff_write_err:
4177 0000D2A6 5E                  <1>         pop    esi ; *
```

```
4178 0000D2A7 BB001C0900          <1>       mov    ebx, FAT_Buffer
4179                              <1>       ; 15/10/2016 (1Dh -> 18)
4180                              <1>       ; 23/03/2016 (1Dh)
4181 0000D2AC B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
4182 0000D2B1 C3                  <1>       retn
4183                              <1>
4184                              <1> calculate_fat_freespace:
4185                              <1>       ; 23/03/2016
4186                              <1>       ; 02/03/2016
4187                              <1>       ; 01/03/2016
4188                              <1>       ; 29/02/2016
4189                              <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
4190                              <1>       ; 30/04/2011
4191                              <1>       ; 03/04/2010
4192                              <1>       ; 2005
4193                              <1>       ; INPUT ->
4194                              <1>       ;      EAX = Cluster count to be added or subtracted
4195                              <1>       ;      If BH = FFh, ESI = TR-DOS Logical Drive Description Table
4196                              <1>       ;      If BH < FFh, BH = TR-DOS Logical Drive Number
4197                              <1>       ;      BL:
4198                              <1>       ;      0 = Calculate, 1 = Add, 2 = Subtract, 3 = Get (Not Set/Calc)
4199                              <1>       ; OUTPUT ->
4200                              <1>       ;      EAX = Free Space in sectors
4201                              <1>       ;      ESI = Logical Dos Drive Description Table address
4202                              <1>       ;      BH = Logical Dos Drive Number (same with input value of BH)
4203                              <1>       ;      BL = Type of operation (same with input value of BL)
4204                              <1>       ;      ECX = 0 -> valid
4205                              <1>       ;      ECX > 0 -> error or invalid
4206                              <1>       ;      If EAX = FFFFFFFFh, it is 're-calculation needed'
4207                              <1>       ;                              sign due to r/w error
4208                              <1>
4209 0000D2B2 66891D[6E930100]    <1>       mov    [CFS_OPType], bx
4210 0000D2B9 A3[70930100]        <1>       mov    [CFS_CC], eax
4211                              <1>
4212 0000D2BE 80FFFF              <1>       cmp    bh, 0FFh
4213 0000D2C1 740B                <1>       je     short pass_calculate_freespace_get_drive_dt_offset
4214                              <1>
4215                              <1> loc_calculate_freespace_get_drive_dt_offset:
4216 0000D2C3 31C0                <1>       xor    eax, eax
4217 0000D2C5 88FC                <1>         mov     ah, bh
4218 0000D2C7 BE00010900          <1>       mov    esi, Logical_DOSDisks
4219 0000D2CC 01C6                <1>         add     esi, eax
4220                              <1>
4221                              <1> pass_calculate_freespace_get_drive_dt_offset:
4222 0000D2CE 08DB                <1>       or     bl, bl
4223 0000D2D0 7435                <1>       jz     short loc_reset_fcc
4224                              <1>
4225                              <1> loc_get_free_sectors:
4226 0000D2D2 8B4674              <1>       mov    eax, [esi+LD_FreeSectors]
4227                              <1>
4228                              <1>       ;xor   ecx, ecx
4229                              <1>       ;dec   ecx ; 0FFFFFFFFh
4230                              <1>       ;cmp   eax, ecx ; 29/02/2016
4231                              <1>       ;je    short loc_get_free_sectors_retn ; recalculation is needed!
4232                              <1>
4233                              <1>       ; 23/03/2016
4234 0000D2D5 8B4E70              <1>       mov    ecx, [esi+LD_TotalSectors]
4235 0000D2D8 39C1                <1>       cmp    ecx, eax ; Total sectors must be greater than Free sectors !
4236 0000D2DA 7707                <1>       ja     short loc_get_free_sectors_check_optype
4237                              <1>
4238 0000D2DC 31C0                <1>       xor    eax, eax
4239 0000D2DE 48                  <1>       dec    eax ; 0FFFFFFFFh  ; recalculation is needed!
4240 0000D2DF 894674              <1>       mov    [esi+LD_FreeSectors], eax ; reset (for recalculation)
4241                              <1>
4242                              <1> loc_get_free_sectors_retn:
4243 0000D2E2 C3                  <1>       retn
4244                              <1>
4245                              <1> loc_get_free_sectors_check_optype:
4246 0000D2E3 80FB03              <1>       cmp    bl, 3
4247 0000D2E6 7203                <1>       jb     short loc_set_fcc
4248                              <1>
4249 0000D2E8 29C9                <1>       sub    ecx, ecx ; 0
4250                              <1>
4251 0000D2EA C3                  <1>       retn
4252                              <1>
4253                              <1> loc_set_fcc:
4254 0000D2EB 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
4255 0000D2EF 0F87DF000000        <1>         ja      loc_update_FAT32_fs_info_fcc
4256                              <1>
4257                              <1>       ;mov   eax, [esi+LD_FreeSectors]
4258 0000D2F5 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
4259 0000D2F9 29D2                <1>       sub    edx, edx
4260 0000D2FB F7F1                <1>       div    ecx
4261                              <1>       ;or    dx, dx
4262                              <1>       ;      ; DX -> Remain sectors < SecPerClust
4263                              <1>       ;      ; DX > 0 -> invalid free sector count
4264                              <1>       ;jnz   short loc_reset_fcc
4265                              <1>
4266                              <1> ;pass_set_fcc_div32:
4267 0000D2FD A3[E7900100]        <1>       mov    [FreeClusterCount], eax
4268 0000D302 E988000000          <1>         jmp     loc_set_free_sectors_FAT12_FAT16
4269                              <1>
4270                              <1> loc_reset_fcc:
4271 0000D307 31C0                <1>       xor    eax, eax
4272 0000D309 A3[E7900100]        <1>       mov    [FreeClusterCount], eax ; 0
4273 0000D30E 8B5678              <1>       mov    edx, [esi+LD_Clusters]
4274 0000D311 42                  <1>       inc    edx
4275 0000D312 8915[D6900100]      <1>       mov    [LastCluster], edx
4276                              <1>
4277 0000D318 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
4278 0000D31C 7647                <1>       jna    short loc_count_free_fat_clusters_0
4279                              <1>
4280 0000D31E 48                  <1>       dec    eax ; FFFFFFFFh
4281 0000D31F A3[78930100]        <1>       mov    [CFS_FAT32FC], eax
4282                              <1>
```

```
                                <1>       ; 29/02/2016
0000D324 89463A                 <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; reset
0000D327 89463E                 <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; reset
                                <1>
0000D32A B802000000             <1>       mov    eax, 2
                                <1>
                                <1> loc_count_fc_next_cluster_0:
0000D32F 50                     <1>       push   eax
0000D330 E801F9FFFF             <1>       call   get_next_cluster
0000D335 7310                   <1>       jnc    short loc_check_fat32_ff_cluster
0000D337 09C0                   <1>       or     eax, eax
0000D339 741E                   <1>       jz     short pass_inc_cfs_fcc_0
                                <1>
                                <1> loc_put_fcc_unknown_sign:
0000D33B 58                     <1>       pop    eax
                                <1>       ; "Free count is Unknown" sign
                                <1>       ;mov   dword [FreeClusterCount], 0FFFFFFFFh
                                <1>
                                <1>       ; 29/02/2016
                                <1>       ; Save Free Cluster Count value in FAT32 'BPB_Reserved' area
                                <1>       ;mov   [esi+LD_BPB+BPB_Reserved], 0FFFFFFFFh ; unknown!
0000D33C 8B15[78930100]         <1>       mov    edx, [CFS_FAT32FC] ; First Free Cluster
                                <1>       ; Save First Free Cluster value in FAT32 'BPB_Reserved+4' area
0000D342 89563E                 <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
                                <1>
0000D345 EB7D                   <1>      jmp     loc_put_fcc_invalid_sign
                                <1>
                                <1> loc_check_fat32_ff_cluster:
0000D347 09C0                   <1>       or     eax, eax
0000D349 750E                   <1>       jnz    short pass_inc_cfs_fcc_0
0000D34B 58                     <1>       pop    eax
0000D34C A3[78930100]           <1>       mov    [CFS_FAT32FC], eax
                                <1>       ;mov   dword [FreeClusterCount], 1
0000D351 FF05[E7900100]         <1>       inc    dword [FreeClusterCount]
0000D357 EB27                   <1>       jmp    short pass_inc_cfs_fcc_1
                                <1>
                                <1> pass_inc_cfs_fcc_0:
0000D359 58                     <1>       pop    eax
                                <1>
                                <1> pass_inc_cfs_fcc_0c:
0000D35A 40                     <1>       inc    eax ; add eax, 1
0000D35B 3B05[D6900100]         <1>       cmp    eax, [LastCluster]
0000D361 76CC                   <1>       jna    short loc_count_fc_next_cluster_0
0000D363 EB6F                   <1>       jmp    short loc_update_FAT32_fs_info_fcc
                                <1>
                                <1> loc_count_free_fat_clusters_0:
                                <1>       ;mov   eax, 2
0000D365 B002                   <1>       mov    al, 2
                                <1>
                                <1> loc_count_fc_next_cluster:
0000D367 50                     <1>       push   eax
0000D368 E8C9F8FFFF             <1>       call   get_next_cluster
0000D36D 720C                   <1>       jc     short loc_count_fcc_stc
                                <1>
                                <1> loc_count_free_clusters_1:
0000D36F 21C0                   <1>       and    eax, eax
0000D371 750C                   <1>       jnz    short pass_inc_cfs_fcc
                                <1>
0000D373 FF05[E7900100]         <1>       inc    dword [FreeClusterCount]
0000D379 EB04                   <1>       jmp    short pass_inc_cfs_fcc
                                <1>
                                <1> loc_count_fcc_stc:
0000D37B 09C0                   <1>       or     eax, eax
0000D37D 75BC                   <1>       jnz    short loc_put_fcc_unknown_sign ; 29/02/2016
                                <1>
                                <1> pass_inc_cfs_fcc:
0000D37F 58                     <1>       pop    eax
                                <1>
                                <1> pass_inc_cfs_fcc_1:
0000D380 40                     <1>       inc    eax ; add eax, 1
0000D381 3B05[D6900100]         <1>       cmp    eax, [LastCluster]
0000D387 76DE                   <1>       jna    short loc_count_fc_next_cluster
                                <1>
                                <1> loc_set_free_sectors:
0000D389 807E0302               <1>       cmp    byte [esi+LD_FATType], 2
0000D38D 7745                   <1>       ja     short loc_update_FAT32_fs_info_fcc
                                <1>
                                <1> loc_set_free_sectors_FAT12_FAT16:
0000D38F 803D[6E930100]00       <1>       cmp    byte [CFS_OPType], 0
0000D396 761C                   <1>       jna    short pass_FAT_add_sub_fcc
0000D398 A1[70930100]           <1>       mov    eax, [CFS_CC]
0000D39D 803D[6E930100]01       <1>       cmp    byte [CFS_OPType], 1
0000D3A4 7708                   <1>       ja     short pass_FAT_add_fcc
0000D3A6 0105[E7900100]         <1>       add    [FreeClusterCount], eax
0000D3AC EB06                   <1>       jmp    short pass_FAT_add_sub_fcc
                                <1>
                                <1> pass_FAT_add_fcc:
0000D3AE 2905[E7900100]         <1>       sub    [FreeClusterCount], eax
                                <1>
                                <1> pass_FAT_add_sub_fcc:
0000D3B4 0FB64613               <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
0000D3B8 8B15[E7900100]         <1>       mov    edx, [FreeClusterCount]
0000D3BE F7E2                   <1>       mul    edx
                                <1>
0000D3C0 31C9                   <1>       xor    ecx, ecx
0000D3C2 EB05                   <1>       jmp    short loc_cfs_retn_params
                                <1>
                                <1> loc_put_fcc_invalid_sign:
0000D3C4 29C0                   <1>              sub    eax, eax ; 0
0000D3C6 48                     <1>       dec    eax ; FFFFFFFFh
                                <1> loc_fat32_ffc_recalc_needed:
0000D3C7 89C1                   <1>       mov    ecx, eax
                                <1>
                                <1> loc_cfs_retn_params:
0000D3C9 894674                 <1>       mov    [esi+LD_FreeSectors], eax
```

```
4388 0000D3CC 0FB71D[6E930100]    <1>       movzx  ebx, word [CFS_OPType]
4389 0000D3D3 C3                  <1>       retn
4390                              <1>
4391                              <1> loc_update_FAT32_fs_info_fcc:
4392                              <1> loc_check_fcc_FSINFO_op:
4393                              <1>       ; 29/02/2016
4394                              <1>       ; EAX = Free cluster count (before this update) ; value from disk
4395                              <1>       ; EDX = First Free Cluster (before this update) ; value from disk
4396 0000D3D4 803D[6E930100]01    <1>       cmp    byte [CFS_OPType], 1
4397 0000D3DB 7221                <1>       jb     short loc_cfs_FAT32_get_rcalc_parms ; 0 = recalculated
4398 0000D3DD 7406                <1>       je     short loc_check_fcc_FSINFO_op1 ; 1 = add
4399                              <1> loc_check_fcc_FSINFO_op2: ; subtract
4400 0000D3DF F71D[70930100]      <1>       neg    dword [CFS_CC] ; prepare to subtract ; 2 = sub (add negative)
4401                              <1> loc_check_fcc_FSINFO_op1:
4402                              <1>       ; 01/03/2016
4403 0000D3E5 31D2                <1>       xor    edx, edx ; 0
4404 0000D3E7 4A                  <1>       dec    edx ; 0FFFFFFFFh
4405 0000D3E8 8B463A              <1>       mov    eax, [esi+LD_BPB+BPB_Reserved]
4406 0000D3EB 39D0                <1>       cmp    eax, edx
4407 0000D3ED 73D5                <1>       jnb    short loc_put_fcc_invalid_sign
4408 0000D3EF 0305[70930100]      <1>       add    eax, [CFS_CC] ; free cluster count on disk + current count
4409 0000D3F5 72CD                <1>       jc     short loc_put_fcc_invalid_sign
4410                              <1>
4411 0000D3F7 A3[E7900100]        <1>       mov    [FreeClusterCount], eax
4412 0000D3FC EB0E                <1>       jmp    short loc_cfs_write_FSINFO_sector
4413                              <1>
4414                              <1> loc_cfs_FAT32_get_rcalc_parms:
4415 0000D3FE 8B15[78930100]      <1>       mov    edx, [CFS_FAT32FC]
4416 0000D404 A1[E7900100]        <1>       mov    eax, [FreeClusterCount]
4417 0000D409 89563E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
4418                              <1> loc_cfs_write_FSINFO_sector:
4419 0000D40C 89463A              <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
4420                              <1>       ; 01/03/2016
4421 0000D40F E8AA000000          <1>       call   set_fat32_fsinfo_sector_parms
4422 0000D414 72AE                <1>       jc     short loc_put_fcc_invalid_sign
4423                              <1>
4424                              <1> loc_set_FAT32_free_sectors:
4425                              <1>       ; 29/02/2016
4426                              <1>       ;mov   eax, [FreeClusterCount]
4427                              <1>       ;mov   ecx, eax
4428                              <1>       ;cmp   eax, 0FFFFFFFFh ; Invalid !
4429                              <1>       ;je    short loc_cfs_retn_params
4430                              <1>       ;
4431 0000D416 8B0D[E7900100]      <1>       mov    ecx, [FreeClusterCount]
4432 0000D41C 0FB64613            <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
4433 0000D420 F7E1                <1>       mul    ecx
4434                              <1>       ; 29/02/2016
4435 0000D422 31C9                <1>       xor    ecx, ecx ; 0
4436 0000D424 09D2                <1>       or     edx, edx ; 0 ?
4437 0000D426 759C                <1>       jnz    loc_put_fcc_invalid_sign
4438 0000D428 394670              <1>       cmp    [esi+LD_TotalSectors], eax ; Volume size in sectors
4439 0000D42B 7697                <1>       jna    short loc_put_fcc_invalid_sign
4440                              <1>       ;
4441                              <1> loc_set_FAT32_free_sectors_ok:
4442 0000D42D 31D2                <1>       xor    edx, edx ; 0
4443 0000D42F EB98                <1>       jmp    short loc_cfs_retn_params
4444                              <1>       ;
4445                              <1>
4446                              <1> get_last_cluster:
4447                              <1>       ; 22/10/2016
4448                              <1>       ; 27/02/2016 (TRDOS 386 =  TRDOS v2.0)
4449                              <1>       ; 12/06/2010 (DRV_FAT.ASM, 'proc_get_last_custer')
4450                              <1>       ; 06/06/2010
4451                              <1>       ; INPUT ->
4452                              <1>       ;      EAX = First Cluster Number
4453                              <1>       ;      ESI = Logical Dos Drive Parameters Table
4454                              <1>       ; OUTPUT ->
4455                              <1>       ;      cf = 0 -> No Error, EAX is valid
4456                              <1>       ;      cf = 1 -> EAX > 0 -> Error
4457                              <1>       ;      EAX = Last Cluster Number
4458                              <1>       ;        ECX = Previous Cluster -just before the last cluster-
4459                              <1>       ;        ; 22/10/2016
4460                              <1>       ;      [glc_index] = cluster index number of the last cluster
4461                              <1>       ;
4462                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
4463                              <1>
4464 0000D431 89C1                <1>       mov    ecx, eax
4465                              <1>
4466 0000D433 C705[80930100]FFFF- <1>       mov    dword [glc_index], 0FFFFFFFFh ; 22/10/2016
4466 0000D43B FFFF                <1>
4467                              <1>
4468                              <1> loc_glc_get_next_cluster_1:
4469 0000D43D 890D[7C930100]      <1>       mov    [glc_prevcluster], ecx
4470                              <1>       ; 22/10/2016
4471 0000D443 FF05[80930100]      <1>       inc    dword [glc_index]
4472                              <1>
4473                              <1> loc_glc_get_next_cluster_2:
4474 0000D449 E8E8F7FFFF          <1>       call   get_next_cluster
4475                              <1>       ; ecx = current/previous cluster
4476                              <1>       ; eax = next/last cluster
4477 0000D44E 73ED                <1>       jnc    short loc_glc_get_next_cluster_1
4478                              <1>
4479 0000D450 09C0                <1>       or     eax, eax
4480 0000D452 7509                <1>       jnz    short loc_glc_stc_retn
4481                              <1>
4482                              <1>       ; ecx = previous cluster
4483 0000D454 89C8                <1>         mov  eax, ecx
4484                              <1>
4485                              <1>       ; previous cluster becomes last cluster (ecx -> eax)
4486                              <1>       ; previous of previous cluster becomes previous cluster (ecx)
4487                              <1>
4488                              <1> loc_glc_prev_cluster_retn:
4489 0000D456 8B0D[7C930100]      <1>       mov    ecx, [glc_prevcluster]
4490 0000D45C C3                  <1>       retn
4491                              <1>
```

```
4492                                <1> loc_glc_stc_retn:
4493 0000D45D F5                    <1>         cmc    ;stc
4494 0000D45E EBF6                  <1>         jmp    short loc_glc_prev_cluster_retn
4495                                <1>
4496                                <1> truncate_cluster_chain:
4497                                <1>         ; 01/03/2016
4498                                <1>         ; 28/02/2016 (TRDOS 386 =  TRDOS v2.0)
4499                                <1>         ; 22/01/2011 (DRV_FAT.ASM, 'proc_truncate_cluster_chain')
4500                                <1>         ; 11/09/2010
4501                                <1>         ; INPUT ->
4502                                <1>         ;      ESI = Logical dos drive description table address
4503                                <1>         ;      EAX = First cluster to be truncated/unlinked
4504                                <1>         ; OUTPUT ->
4505                                <1>         ;      ESI = Logical dos drive description table address
4506                                <1>         ;      ECX = Count of truncated/removed clusters
4507                                <1>         ;      CF = 0 -> EAX = Free sectors
4508                                <1>         ;      CF = 1 -> Error code in EAX (AL)
4509                                <1>
4510                                <1>         ; NOTE: This procedure does not update lm date&time !
4511                                <1>
4512                                <1> loc_truncate_cc:
4513 0000D460 31C9                  <1>         xor    ecx, ecx ; mov ecx, 0
4514                                <1>         ;mov   byte [FAT_BuffValidData], 0
4515 0000D462 890D[D2900100]        <1>         mov    [FAT_ClusterCounter], ecx ; 0 ; reset
4516                                <1>
4517                                <1> loc_tcc_unlink_clusters:
4518 0000D468 E8F3FAFFFF            <1>         call   update_cluster
4519                                <1>         ; EAX = Next Cluster
4520                                <1>         ; ECX = Cluster Value
4521                                <1>         ; Note:
4522                                <1>         ; Returns count of unlinked clusters in
4523                                <1>         ; dword ptr FAT_ClusterCounter
4524 0000D46D 73F9                  <1>         jnc short loc_tcc_unlink_clusters
4525                                <1>
4526                                <1> pass_tcc_unlink_clusters:
4527 0000D46F A2[87930100]          <1>         mov    byte [TCC_FATErr], al
4528 0000D474 803D[CA900100]02      <1>         cmp    byte [FAT_BuffValidData], 2
4529 0000D47B 750E                  <1>         jne    short loc_tcc_calculate_FAT_freespace
4530 0000D47D E89BFDFFFF            <1>         call   save_fat_buffer
4531 0000D482 7307                  <1>         jnc    short loc_tcc_calculate_FAT_freespace
4532 0000D484 A2[87930100]          <1>         mov    byte [TCC_FATErr], al ; Error
4533                                <1>         ;mov   byte [FAT_BuffValidData], 0
4534                                <1>
4535                                <1>         ; 01/03/2016
4536 0000D489 EB12                  <1>         jmp    short loc_tcc_recalculate_FAT_freespace
4537                                <1>
4538                                <1> loc_tcc_calculate_FAT_freespace:
4539 0000D48B A1[D2900100]          <1>         mov    eax, [FAT_ClusterCounter] ; signed (+-) number
4540 0000D490 66BB01FF              <1>         mov    bx, 0FF01h ; BH = FFh -> ESI = Dos drv desc. table
4541                                <1>                           ; BL = 1 -> add cluster
4542 0000D494 E819FEFFFF            <1>         call   calculate_fat_freespace
4543 0000D499 21C9                  <1>         and    ecx, ecx ; cx = 0 -> valid free sector count
4544 0000D49B 7409                  <1>         jz     short pass_truncate_cc_recalc_FAT_freespace
4545                                <1>
4546                                <1> loc_tcc_recalculate_FAT_freespace:
4547 0000D49D 66BB00FF              <1>         mov    bx, 0FF00h ; recalculate !
4548 0000D4A1 E80CFEFFFF            <1>         call   calculate_fat_freespace
4549                                <1>
4550                                <1> loc_tcc_calculate_FAT_freespace_err:
4551                                <1> pass_truncate_cc_recalc_FAT_freespace:
4552 0000D4A6 8B0D[D2900100]        <1>         mov    ecx, [FAT_ClusterCounter]
4553                                <1>
4554 0000D4AC 803D[87930100]00      <1>         cmp    byte [TCC_FATErr], 0
4555 0000D4B3 7608                  <1>         jna    short loc_tcc_unlink_clusters_retn
4556                                <1>
4557                                <1> loc_tcc_unlink_clusters_error:
4558 0000D4B5 0FB605[87930100]      <1>         movzx  eax, byte [TCC_FATErr]
4559 0000D4BC F9                    <1>         stc
4560                                <1> loc_tcc_unlink_clusters_retn:
4561 0000D4BD C3                    <1>         retn
4562                                <1>
4563                                <1> set_fat32_fsinfo_sector_parms:
4564                                <1>         ; 15/10/2016
4565                                <1>         ; 23/03/2016
4566                                <1>         ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
4567                                <1>         ; INPUT ->
4568                                <1>         ;      ESI = Logical dos drive description table address
4569                                <1>         ;      [esi+LD_BPB+BPB_Reserved] = Free Cluster Count
4570                                <1>         ;      [esi+LD_BPB+BPB_Reserved+4] = First Free Cluster
4571                                <1>         ; OUTPUT ->
4572                                <1>         ;      ESI = Logical dos drive description table address
4573                                <1>         ;      CF = 0 -> OK..
4574                                <1>         ;      CF = 1 -> Error code in EAX (AL)
4575                                <1>         ;
4576                                <1>         ; (Modified registers: EAX, EBX, ECX, EDX)
4577                                <1>
4578 0000D4BE E824000000            <1>         call   get_fat32_fsinfo_sector_parms
4579 0000D4C3 7221                  <1>         jc     short update_fat32_fsinfo_sector_retn
4580                                <1>
4581 0000D4C5 8B463A                <1>         mov    eax, [esi+LD_BPB+BPB_Reserved] ; Free Cluster Count
4582 0000D4C8 8B563E                <1>         mov    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free Cluster
4583                                <1>
4584                                <1>         ;mov   ebx, DOSBootSectorBuff
4585 0000D4CB 8983E8010000          <1>         mov    [ebx+488], eax
4586 0000D4D1 8993EC010000          <1>         mov    [ebx+492], edx
4587                                <1>
4588 0000D4D7 A1[74930100]          <1>         mov    eax, [CFS_FAT32FSINFOSEC]
4589 0000D4DC B901000000            <1>         mov    ecx, 1
4590 0000D4E1 E888550000            <1>         call   disk_write
4591                                <1>         ;jnc   short update_fat32_fsinfo_sector_retn
4592                                <1>
4593                                <1>         ; 15/10/2016 (1Dh -> 18)
4594                                <1>         ; 23/03/2016 (1Dh)
4595                                <1>         ;mov   eax, 18 ; Drive not ready or write error
4596                                <1>
```

```
                                <1> update_fat32_fsinfo_sector_retn:
0000D4E6 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>
0000D4E7 0FB74636               <1>     movzx  eax, word [esi+LD_BPB+FAT32_FSInfoSec]
0000D4EB 03466C                 <1>     add    eax, [esi+LD_StartSector]
0000D4EE A3[74930100]           <1>     mov    [CFS_FAT32FSINFOSEC], eax
                                <1>
0000D4F3 BB[C68E0100]           <1>     mov   ebx, DOSBootSectorBuff
0000D4F8 B901000000             <1>     mov    ecx, 1
0000D4FD E87B550000             <1>     call   disk_read
0000D502 7232                   <1>     jc     short loc_read_FAT32_fsinfo_sec_err
                                <1>
0000D504 BB[C68E0100]           <1>     mov    ebx, DOSBootSectorBuff
                                <1>
0000D509 813B52526141           <1>     cmp    dword [ebx], 41615252h
0000D50F 751E                   <1>     jne    short loc_read_FAT32_fsinfo_sec_stc
                                <1>
0000D511 81BBE4010000727241-    <1>     cmp    dword [ebx+484], 61417272h
0000D51A 61                     <1>
0000D51B 7512                   <1>     jne    short loc_read_FAT32_fsinfo_sec_stc
                                <1>
0000D51D A1[74930100]           <1>     mov    eax, [CFS_FAT32FSINFOSEC]
0000D522 8B8BE8010000           <1>     mov    ecx, [ebx+488] ; free cluster count
0000D528 8B93EC010000           <1>     mov    edx, [ebx+492] ; first (next) free cluster
                                <1>
0000D52E C3                     <1>     retn
                                <1>
                                <1> loc_read_FAT32_fsinfo_sec_stc:
                                <1>     ; 15/10/2016 (0Bh -> 28)
0000D52F B81C000000             <1>     mov    eax, 28 ; Invalid format!
0000D534 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)
0000D536 B811000000             <1>     mov    eax, 17 ; Drive not ready or read error
                                <1>
                                <1> loc_read_FAT32_fsinfo_sec_stc_retn:
0000D53B 29DB                   <1>     sub    ebx, ebx ; 0
0000D53D F9                     <1>     stc
0000D53E 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>
0000D53F A3[A4940100]           <1>     mov    [FAT_anc_LCluster], eax
                                <1>
0000D544 E844F9FFFF             <1>     call   get_first_free_cluster
0000D549 720B                   <1>     jc     short loc_add_new_cluster_retn
                                <1>     ; EAX >= 2 and EAX < FFFFFFFFh is valid
                                <1>
0000D54B 89C2                   <1>     mov    edx, eax
                                <1>
0000D54D 42                     <1>     inc    edx
                                <1>     ;jnz   short loc_add_new_cluster_check_ffc_eax
0000D54E 7516                   <1>     jnz    short loc_add_new_cluster_save_fcc
                                <1>
                                <1> loc_add_new_cluster_no_disk_space_retn:
0000D550 B827000000             <1>     mov    eax, 27h ; MSDOS err => insufficient disk space
                                <1> loc_add_new_cluster_stc_retn:
0000D555 F9                     <1>     stc
                                <1> loc_add_new_cluster_retn:
0000D556 0FB65E13               <1>     movzx  ebx, byte [esi+LD_BPB+SecPerClust]
0000D55A 8B4E74                 <1>     mov    ecx, [esi+LD_FreeSectors]
                                <1>     ;xor   edx, edx
```

```
                                <1>       ;stc
0000D55D C3                     <1>       retn
                                <1>
                                <1> loc_anc_invalid_format_stc_retn:
0000D55E F9                     <1>       stc
                                <1> loc_add_new_cluster_invalid_format_retn:
                                <1>       ; 15/10/2016 (0Bh -> 28)
0000D55F B81C000000             <1>       mov    eax, 28 ; Invalid format
0000D564 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:
0000D566 A3[A8940100]           <1>       mov    [FAT_anc_FFCluster], eax
                                <1>
0000D56B 83E802                 <1>       sub    eax, 2
0000D56E 0FB65E13               <1>       movzx ebx, byte [esi+LD_BPB+SecPerClust]
0000D572 F7E3                   <1>       mul    ebx
0000D574 09D2                   <1>       or     edx, edx
0000D576 75E6                   <1>       jnz    short loc_anc_invalid_format_stc_retn
                                <1>
                                <1> loc_add_new_cluster_allocate_cluster:
                                <1>       ; 18/03/2016
0000D578 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
0000D579 BF00000700             <1>       mov    edi, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
0000D57E 89D9                   <1>       mov    ecx, ebx ; sector count
0000D580 C1E107                 <1>       shl    ecx, 7 ; 1 sector = 512 bytes -> 128 double words
                                <1>       ;xor   eax, eax ; 0
0000D583 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
0000D585 89D0                   <1>       mov    eax, edx
0000D587 034668                 <1>         add     eax, [esi+LD_DATABegin]
0000D58A 72D3                   <1>       jc     short loc_add_new_cluster_invalid_format_retn
                                <1>
0000D58C 89D9                   <1>       mov    ecx, ebx ; ECX = sectors per cluster (<256)
0000D58E BB00000700             <1>       mov    ebx, Cluster_Buffer
0000D593 E8D6540000             <1>       call   disk_write
0000D598 7307                   <1>       jnc    short loc_add_new_cluster_update_fat_nlc
                                <1>
                                <1>       ; 15/10/2016 (1Dh -> 18)
0000D59A B812000000             <1>       mov    eax, 18 ; Write Error
0000D59F EBB4                   <1>       jmp    short loc_add_new_cluster_stc_retn
                                <1>
                                <1> loc_add_new_cluster_update_fat_nlc:
0000D5A1 A1[A8940100]           <1>       mov    eax, [FAT_anc_FFCluster]
0000D5A6 31C9                   <1>       xor    ecx, ecx
0000D5A8 890D[D2900100]         <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; reset
0000D5AE 49                     <1>       dec    ecx ; 0FFFFFFFFh
0000D5AF E8ACF9FFFF             <1>       call   update_cluster
0000D5B4 7304                   <1>       jnc    short loc_add_new_cluster_update_fat_plc
0000D5B6 09C0                   <1>       or     eax, eax ;EAX = 0 -> cluster value is 0 or eocc
0000D5B8 759B                   <1>       jnz    short loc_add_new_cluster_stc_retn
                                <1>
                                <1> loc_add_new_cluster_update_fat_plc:
0000D5BA A1[A4940100]           <1>       mov    eax, [FAT_anc_LCluster]
0000D5BF 8B0D[A8940100]         <1>       mov    ecx, [FAT_anc_FFCluster]
0000D5C5 E896F9FFFF             <1>       call   update_cluster
0000D5CA 7314                   <1>       jnc    short loc_add_new_cluster_save_fat_buffer
0000D5CC 09C0                   <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
0000D5CE 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>
0000D5D0 66BB00FF               <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
                                <1>                          ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
0000D5D4 50                     <1>       push   eax
0000D5D5 E8D8FCFFFF             <1>       call   calculate_fat_freespace
0000D5DA 58                     <1>       pop    eax
0000D5DB 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
0000D5E0 E838FCFFFF             <1>       call   save_fat_buffer
0000D5E5 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
0000D5E7 A1[D2900100]           <1>       mov    eax, [FAT_ClusterCounter]
0000D5EC 66BB01FF               <1>       mov    bx, 0FF01h ; BH = FFh -> ESI -> Dos drv desc. table
                                <1>              ; BL = 1 -> add cluster
0000D5F0 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)
0000D5F2 E8BBFCFFFF             <1>         call    calculate_fat_freespace
                                <1>       ;ECX = 0 -> no error, ECX > 0 -> error or invalid return
```

```
4806 0000D5F7 21C9              <1>       and    ecx, ecx ; ECX = 0 -> valid free sector count
4807 0000D5F9 7409              <1>       jz     short loc_add_new_cluster_return_cluster_number
4808                            <1>
4809                            <1> loc_add_new_cluster_recalc_FAT_freespace:
4810 0000D5FB 66BB00FF          <1>       mov    bx, 0FF00h  ; recalculate free space
4811 0000D5FF E8AEFCFFFF        <1>         call    calculate_fat_freespace
4812                            <1>       ; cf = 0
4813                            <1> loc_add_new_cluster_return_cluster_number:
4814 0000D604 89C1              <1>       mov    ecx, eax ; Free sector count
4815 0000D606 A1[A8940100]      <1>       mov    eax, [FAT_anc_FFCluster]
4816 0000D60B 0FB65E13          <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
4817                            <1>       ;mov   edi, Cluster_Buffer
4818 0000D60F 31D2              <1>       xor    edx, edx
4819 0000D611 C3                <1>       retn
4820                            <1>
4821                            <1> write_cluster:
4822                            <1>       ; 15/10/2016
4823                            <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
4824                            <1>       ;
4825                            <1>       ; INPUT ->
4826                            <1>       ;      EAX = Cluster Number (Sector index for SINGLIX FS)
4827                            <1>       ;      ESI = Logical DOS Drive Description Table address
4828                            <1>       ;      EBX = Cluster (File R/W) Buffer address (max. 64KB)
4829                            <1>       ;      Only for SINGLIX FS:
4830                            <1>       ;      EDX = File Number (The 1st FDT address)
4831                            <1>       ; OUTPUT ->
4832                            <1>       ;      cf = 1 -> Cluster can not be written onto disk
4833                            <1>       ;          EAX > 0 -> Error number
4834                            <1>       ;      cf = 0 -> Cluster has been written successfully
4835                            <1>       ;
4836                            <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
4837                            <1>
4838 0000D612 0FB64E13          <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
4839                            <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
4840                            <1>
4841                            <1> write_file_sectors: ; 16/03/2016
4842 0000D616 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
4843 0000D61A 761C              <1>       jna    short write_fs_cluster
4844                            <1>
4845                            <1> write_fat_file_sectors:
4846 0000D61C 83E802            <1>       sub    eax, 2 ; Beginning cluster number is always 2
4847 0000D61F 0FB65613          <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
4848 0000D623 F7E2              <1>       mul    edx
4849 0000D625 034668            <1>       add    eax, [esi+LD_DATABegin] ; absolute address of the cluster
4850                            <1>
4851                            <1>       ; EAX = Disk sector address
4852                            <1>       ; ECX = Sector count
4853                            <1>       ; EBX = Buffer address
4854                            <1>       ; (EDX = 0)
4855                            <1>       ; ESI = Logical DOS drive description table address
4856                            <1>
4857 0000D628 E841540000        <1>       call   disk_write
4858 0000D62D 7306              <1>       jnc    short wclust_retn
4859                            <1>
4860                            <1>       ; 15/10/2016 (1Dh -> 18)
4861 0000D62F B812000000        <1>       mov    eax, 18 ; Drive not ready or write error !
4862 0000D634 C3                <1>       retn
4863                            <1>
4864                            <1> wclust_retn:
4865 0000D635 29C0              <1>       sub    eax, eax ; 0
4866 0000D637 C3                <1>       retn
4867                            <1>
4868                            <1> write_fs_cluster:
4869                            <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
4870                            <1>       ; Singlix FS
4871                            <1>
4872                            <1>       ; EAX = Cluster number is sector index number of the file (eax)
4873                            <1>
4874                            <1>       ; EDX = File number is the first File Descriptor Table address
4875                            <1>       ;      of the file. (Absolute address of the FDT).
4876                            <1>
4877                            <1>       ; eax = sector index (0 for the first sector)
4878                            <1>       ; edx = FDT0 address
4879                            <1>             ; 64 KB buffer = 128 sectors (limit)
4880 0000D638 B980000000        <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
4881 0000D63D E801000000        <1>       call   write_fs_sectors
4882 0000D642 C3                <1>       retn
4883                            <1>
4884                            <1> write_fs_sectors:
4885                            <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
4886 0000D643 F9                <1>       stc
4887 0000D644 C3                <1>       retn
4888                            <1>
4889                            <1> get_cluster_by_index:
4890                            <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
4891                            <1>       ; INPUT ->
4892                            <1>       ;      EAX = Beginning cluster
4893                            <1>       ;      EDX = Sector index in disk/file section
4894                            <1>       ;            (Only for SINGLIX file system!)
4895                            <1>       ;      ECX = Cluster sequence number after the beginning cluster
4896                            <1>       ;      ESI = Logical DOS Drive Description Table address
4897                            <1>       ; OUTPUT ->
4898                            <1>       ;      EAX = Cluster number
4899                            <1>       ;      cf = 1 -> Error code in AL (EAX)
4900                            <1>       ;
4901                            <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
4902                            <1>       ;
4903 0000D645 807E0301          <1>       cmp    byte [esi+LD_FATType], 1
4904 0000D649 721E              <1>         jb      short get_fs_section_by_index
4905                            <1>
4906 0000D64B 3B4E78            <1>       cmp    ecx, [esi+LD_Clusters]
4907 0000D64E 7207              <1>       jb     short gcbi_1
4908                            <1> gcbi_0:
4909 0000D650 F9                <1>       stc
4910 0000D651 B823000000        <1>       mov    eax, 23h ; Cluster not available !
```

```
4911                                    <1>                     ; MSDOS error code: FCB unavailable
4912 0000D656 C3                        <1>     retn
4913                                    <1> gcbi_1:
4914 0000D657 51                        <1>     push    ecx
4915 0000D658 E8D9F5FFFF                <1>     call    get_next_cluster
4916 0000D65D 59                        <1>     pop     ecx
4917 0000D65E 7203                      <1>     jc      short gcbi_3
4918 0000D660 E2F5                      <1>     loop    gcbi_1
4919                                    <1> gcbi_2:
4920 0000D662 C3                        <1>     retn
4921                                    <1> gcbi_3:
4922 0000D663 09C0                      <1>     or      eax, eax
4923 0000D665 74E9                      <1>     jz      short gcbi_0
4924 0000D667 F5                        <1>     cmc     ; stc
4925 0000D668 C3                        <1>     retn
4926                                    <1>
4927                                    <1> get_fs_section_by_index:
4928                                    <1>     ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
4929                                    <1>     ; INPUT ->
4930                                    <1>     ;       EAX = Beginning FDT number/address
4931                                    <1>     ;       EDX = Sector index in disk/file section
4932                                    <1>     ;       ECX = Sector sequence number after the beginning FDT
4933                                    <1>     ;       ESI = Logical DOS Drive Description Table address
4934                                    <1>     ; OUTPUT ->
4935                                    <1>     ;       EAX = FDT number/address
4936                                    <1>     ;       EDX = Sector index of the section (0,1,2,3,4...)
4937                                    <1>     ;       cf = 1 -> Error code in AL (EAX)
4938                                    <1>     ;
4939                                    <1>     ;(Modified registers: EAX, ECX, EBX, EDX)
4940                                    <1>     ;
4941 0000D669 B8FFFFFFFF                <1>     mov     eax, 0FFFFFFFFh
4942 0000D66E C3                        <1>     retn
4943                                    <1>
4944                                    <1> get_last_section:
4945                                    <1>     ; 22/10/2016 (TRDOS 386 =  TRDOS v2.0)
4946                                    <1>     ; INPUT ->
4947                                    <1>     ;       EAX = (The 1st) FDT number/address
4948                                    <1>     ;       ESI = Logical DOS Drive Description Table address
4949                                    <1>     ; OUTPUT ->
4950                                    <1>     ;       EAX = FDT number/address of the last section
4951                                    <1>     ;       EDX = Last sector of the section (0,1,2,3,4...)
4952                                    <1>     ;       [glc_index] = sector index number of the last sector
4953                                    <1>     ;                     (for file, not for the last section)
4954                                    <1>     ;
4955                                    <1>     ;       cf = 1 -> Error code in AL (EAX)
4956                                    <1>     ;
4957                                    <1>     ;(Modified registers: EAX, ECX, EBX, EDX)
4958                                    <1>     ;
4959 0000D66F B800000000                <1>     mov     eax, 0
4960 0000D674 BA00000000                <1>     mov     edx, 0
4961 0000D679 C3                        <1>     retn
3089                                        %include 'trdosk6.s' ; 24/01/2016
3090                                    <1> ; ****************************************************************************
3091                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.3) - MAIN PROGRAM : trdosk6.s
3092                                    <1> ; ----------------------------------------------------------------------------
3093                                    <1> ; Last Update: 06/03/2021
3094                                    <1> ; ----------------------------------------------------------------------------
3095                                    <1> ; Beginning: 24/01/2016
3096                                    <1> ; ----------------------------------------------------------------------------
3097                                    <1> ; Assembler: NASM version 2.15 t(trdos386.s)
3098                                    <1> ; ----------------------------------------------------------------------------
3099                                    <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3100                                    <1> ; u1.s (27/17/2015), u2.s (03/01/2016)
3101                                    <1> ; ****************************************************************************
3102                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3103                                    <1> ; TRDOS2.ASM (09/11/2011)
3104                                    <1> ; ----------------------------------------------------------------------------
3105                                    <1> ; INT_21H.ASM (c) 2009-2011 Erdogan TAN  [14/11/2009] Last Update: 08/11/2011
3106                                    <1>
3107                                    <1> sysent: ; < enter to system call >
3108                                    <1>     ; 17/03/2017
3109                                    <1>     ; 03/03/2017
3110                                    <1>     ; 19/02/2017
3111                                    <1>     ; 13/01/2017
3112                                    <1>     ; 06/06/2016
3113                                    <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3114                                    <1>     ; 16/04/2015 - 19/10/2015 (Retro UNIX 386 v1)
3115                                    <1>     ; 10/04/2013 - 18/01/2014 (Retro UNIX 8086 v1)
3116                                    <1>     ;
3117                                    <1>     ; 'unkni' or 'sysent' is sytem entry from various traps.
3118                                    <1>     ; The trap type is determined and an indirect jump is made to
3119                                    <1>     ; the appropriate system call handler. If there is a trap inside
3120                                    <1>     ; the system a jump to panic is made. All user registers are saved
3121                                    <1>     ; and u.sp points to the end of the users stack. The sys (trap)
3122                                    <1>     ; instructor is decoded to get the the system code part (see
3123                                    <1>     ; trap instruction in the PDP-11 handbook) and from this
3124                                    <1>     ; the indirect jump address is calculated. If a bad system call is
3125                                    <1>     ; made, i.e., the limits of the jump table are exceeded, 'badsys'
3126                                    <1>     ; is called. If the call is legitimate control passes to the
3127                                    <1>     ; appropriate system routine.
3128                                    <1>     ;
3129                                    <1>     ; Calling sequence:
3130                                    <1>     ;       Through a trap caused by any sys call outside the system.
3131                                    <1>     ; Arguments:
3132                                    <1>     ;       Arguments of particular system call.
3133                                    <1>     ; ..........................................................
3134                                    <1>     ;
3135                                    <1>     ; Retro UNIX 8086 v1 modification:
3136                                    <1>     ;       System call number is in EAX register.
3137                                    <1>     ;
3138                                    <1>     ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
3139                                    <1>     ;       registers depending of function details.
3140                                    <1>     ;
3141                                    <1>     ; 16/04/2015
3142 0000D67A 368925[5C030300]          <1>       mov     [ss:u.sp], esp ; Kernel stack points to return address
```

```
3143                                    <1>
3144                                    <1>       ; save user registers
3145 0000D681 1E                        <1>       push   ds
3146 0000D682 06                        <1>       push   es
3147 0000D683 0FA0                      <1>       push   fs
3148 0000D685 0FA8                      <1>       push   gs
3149 0000D687 60                        <1>       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
3150                                    <1>       ;
3151                                    <1>       ; ESPACE = [ss:u.sp] - esp ; 4*12 = 48 ; 17/09/2015 ; 06/06/2016
3152                                    <1>       ;      (ESPACE is size of space in kernel stack
3153                                    <1>       ;      for saving/restoring user registers.)
3154                                    <1>       ;
3155 0000D688 50                        <1>       push   eax ; 01/07/2015
3156 0000D689 66B81000                  <1>       mov    ax, KDATA
3157 0000D68D 8ED8                      <1>       mov    ds, ax
3158 0000D68F 8EC0                      <1>       mov    es, ax
3159 0000D691 8EE0                      <1>       mov    fs, ax
3160 0000D693 8EE8                      <1>       mov    gs, ax
3161 0000D695 A1[F0880100]              <1>       mov    eax, [k_page_dir]
3162 0000D69A 0F22D8                    <1>       mov    cr3, eax
3163 0000D69D 58                        <1>       pop    eax ; 01/07/2015
3164                                    <1>       ; 19/10/2015
3165 0000D69E FC                        <1>       cld
3166                                    <1>       ;
3167 0000D69F FE05[5B030300]            <1>       inc    byte [sysflg]
3168                                    <1>              ; incb sysflg / indicate a system routine is in progress
3169 0000D6A5 FB                        <1>       sti    ; 18/01/2014
3170 0000D6A6 0F85DF9DFFFF              <1>       jnz     panic ; 24/05/2013
3171                                    <1>              ; beq 1f
3172                                    <1>              ; jmp panic ; / called if trap inside system
3173                                    <1> ;1:
3174                                    <1>       ; 17/03/2017
3175 0000D6AC 80642438FE                <1>       and    byte [esp+ESPACE+8], ~1 ; clear carry flag
3176                                    <1>
3177                                    <1>       ; 16/04/2015
3178 0000D6B1 A3[64030300]              <1>       mov    [u.r0], eax
3179 0000D6B6 8925[60030300]            <1>       mov    [u.usp], esp ; kernel stack points to user's registers
3180                                    <1>
3181                                    <1>       ; 13/01/2017 (TRDOS 386 Feaure only !)
3182 0000D6BC 803D[D4030300]00          <1>       cmp    byte [u.t_lock], 0 ; timer interrupt lock ?
3183 0000D6C3 0F879D010000              <1>       ja     sysrele               ; yes, sys release only !!!
3184                                    <1>
3185                                    <1>              ; mov $s.syst+2,clockp
3186                                    <1>              ; mov r0,-(sp) / save user registers
3187                                    <1>              ; mov sp,u.r0 / pointer to bottom of users stack
3188                                    <1>                       ; / in u.r0
3189                                    <1>              ; mov r1,-(sp)
3190                                    <1>              ; mov r2,-(sp)
3191                                    <1>              ; mov r3,-(sp)
3192                                    <1>              ; mov r4,-(sp)
3193                                    <1>              ; mov r5,-(sp)
3194                                    <1>              ; mov ac,-(sp) / "accumulator" register for extended
3195                                    <1>                             ; / arithmetic unit
3196                                    <1>              ; mov mq,-(sp) / "multiplier quotient" register for the
3197                                    <1>                             ; / extended arithmetic unit
3198                                    <1>              ; mov sc,-(sp) / "step count" register for the extended
3199                                    <1>                             ; / arithmetic unit
3200                                    <1>              ; mov sp,u.sp / u.sp points to top of users stack
3201                                    <1>              ; mov 18.(sp),r0 / store pc in r0
3202                                    <1>              ; mov -(r0),r0 / sys inst in r0      10400xxx
3203                                    <1>              ; sub $sys,r0 / get xxx code
3204 0000D6C9 C1E002                    <1>       shl    eax, 2
3205                                    <1>              ; asl r0 / multiply by 2 to jump indirect in bytes
3206 0000D6CC 3DB8000000                <1>       cmp    eax, end_of_syscalls - syscalls
3207                                    <1>              ; cmp r0,$2f-1f / limit of table (35) exceeded
3208                                    <1>       ;jnb   short badsys
3209                                    <1>              ; bhis badsys / yes, bad system call
3210 0000D6D1 F5                        <1>       cmc
3211 0000D6D2 9C                        <1>       pushf
3212 0000D6D3 50                        <1>       push   eax
3213 0000D6D4 8B2D[5C030300]            <1>       mov    ebp, [u.sp] ; Kernel stack at the beginning of sys call
3214 0000D6DA B0FE                      <1>       mov    al, 0FEh ; 11111110b
3215 0000D6DC 1400                      <1>       adc    al, 0 ; al = al + cf
3216 0000D6DE 204508                    <1>       and    [ebp+8], al ; flags (reset carry flag)
3217                                    <1>              ; bic $341,20.(sp) / set users processor priority to 0
3218                                    <1>                                 ; / and clear carry bit
3219 0000D6E1 5D                        <1>       pop    ebp ; eax
3220 0000D6E2 9D                        <1>       popf
3221 0000D6E3 0F8208020000              <1>       jc     badsys
3222 0000D6E9 A1[64030300]              <1>       mov    eax, [u.r0]
3223                                    <1>       ; system call registers: EAX, EDX, ECX, EBX, ESI, EDI
3224 0000D6EE FFA5[F4D60000]            <1>       jmp    dword [ebp+syscalls]
3225                                    <1>              ; jmp *1f(r0) / jump indirect thru table of addresses
3226                                    <1>                           ; / to proper system routine.
3227                                    <1> syscalls: ; 1:
3228                                    <1>       ; 31/12/2017
3229                                    <1>       ; 28/02/2017
3230                                    <1>       ; 20/02/2017
3231                                    <1>       ; 19/02/2017
3232                                    <1>       ; 15/10/2016
3233                                    <1>       ; 20/05/2016
3234                                    <1>       ; 19/05/2016
3235                                    <1>       ; 16/05/2016
3236                                    <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3237                                    <1>       ; 21/09/2015
3238                                    <1>       ; 01/07/2015
3239                                    <1>       ; 16/04/2015 (32 bit address modification)
3240 0000D6F4 [68190100]                <1>       dd sysver    ; 0 ; Get TRDOS 386 version number (v2.0)
3241 0000D6F8 [53D90000]                <1>       dd sysexit   ; 1
3242 0000D6FC [28DB0000]                <1>       dd sysfork   ; 2
3243 0000D700 [5BDF0000]                <1>       dd sysread   ; 3
3244 0000D704 [7ADF0000]                <1>       dd syswrite  ; 4
3245 0000D708 [11DD0000]                <1>       dd sysopen   ; 5
3246 0000D70C [32DF0000]                <1>       dd sysclose  ; 6
3247 0000D710 [AADA0000]                <1>       dd syswait   ; 7
```

```
3248 0000D714 [40DC0000]          <1>       dd syscreat  ; 8
3249 0000D718 [BC270100]          <1>       dd sysrename ; 9  ; TRDOS 386, Rename File (31/12/2017)
3250 0000D71C [37230100]          <1>       dd sysdelete ; 10 ; TRDOS 386, Delete File (29/12/2017)
3251 0000D720 [050D0100]          <1>       dd sysexec   ; 11
3252 0000D724 [61240100]          <1>       dd syschdir  ; 12
3253 0000D728 [26260100]          <1>       dd systime   ; 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
3254 0000D72C [F4DE0000]          <1>       dd sysmkdir  ; 14
3255 0000D730 [95240100]          <1>       dd syschmod  ; 15 ; TRDOS 386, Change Attributes (30/12/2017)
3256 0000D734 [9E230100]          <1>       dd sysrmdir  ; 16 ; TRDOS 386, Remove Directory (29/12/2017)
3257 0000D738 [E00F0100]          <1>       dd sysbreak  ; 17
3258 0000D73C [7B250100]          <1>       dd sysdrive  ; 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
3259 0000D740 [21100100]          <1>       dd sysseek   ; 19
3260 0000D744 [33100100]          <1>       dd systell   ; 20
3261 0000D748 [DD280100]          <1>       dd sysmem    ; 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
3262 0000D74C [13290100]          <1>       dd sysprompt ; 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
3263 0000D750 [55290100]          <1>       dd syspath   ; 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
3264 0000D754 [C2290100]          <1>       dd sysenv    ; 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
3265 0000D758 [A7260100]          <1>       dd sysstime  ; 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
3266 0000D75C [99100100]          <1>       dd sysquit   ; 26
3267 0000D760 [8D100100]          <1>       dd sysintr   ; 27
3268 0000D764 [CA250100]          <1>       dd sysdir    ; 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
3269 0000D768 [11E00000]          <1>       dd sysemt    ; 29
3270 0000D76C [05260100]          <1>       dd sysldrvt  ; 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
3271 0000D770 [C2E10000]          <1>       dd sysvideo  ; 31 ; TRDOS 386 Video Functions (16/05/2016)
3272 0000D774 [8C330100]          <1>       dd sysaudio  ; 32 ; TRDOS 386 Audio Functions (16/05/2016)
3273 0000D778 [2AE00000]          <1>       dd systimer  ; 33 ; TRDOS 386 Timer Functions (18/05/2016)
3274 0000D77C [DA100100]          <1>       dd syssleep  ; 34 ; Retro UNIX 8086 v1 feature only !
3275                              <1>                          ; 11/06/2014
3276 0000D780 [09110100]          <1>       dd sysmsg    ; 35 ; Retro UNIX 386 v1 feature only !
3277                              <1>                          ; 01/07/2015
3278 0000D784 [26120100]          <1>       dd sysgeterr ; 36 ; Retro UNIX 386 v1 feature only !
3279                              <1>                          ; 21/09/2015 - get last error number
3280 0000D788 [0E230100]          <1>       dd sysfpstat ; 37 ; TRDOS 386 FPU state option (28/02/2017)
3281 0000D78C [77190100]          <1>       dd syspri    ; 38 ; change priority - TRDOS 386 (20/05/2016)
3282 0000D790 [66D80000]          <1>       dd sysrele   ; 39 ; TRDOS 386 (19/05/2016) (0 -> 39)
3283 0000D794 [AA1A0100]          <1>       dd sysfff    ; 40 ; Find First File - TRDOS 386 (15/10/2016)
3284 0000D798 [891B0100]          <1>       dd sysfnf    ; 41 ; Find Next File - TRDOS 386 (15/10/2016)
3285 0000D79C [F9210100]          <1>       dd sysalloc  ; 42 ; Allocate contiguous memory block/pages
3286                              <1>                          ; TRDOS 386 (19/02/2017) DMA buff fuctions
3287 0000D7A0 [B7220100]          <1>       dd sysdalloc ; 43 ; Deallocate contiguous memory block/pages
3288                              <1>                          ; TRDOS 386 (19/02/2017) DMA buff fuctions
3289 0000D7A4 [F2220100]          <1>       dd syscalbac ; 44 ; IRQ Callback and Signal Response Byte
3290                              <1>                          ; service setup - TRDOS 386 (20/02/2017)
3291                              <1>                          ; 28/08/2017 (20/08/2017)
3292 0000D7A8 [1D3C0100]          <1>       dd sysdma    ; 45 ; TRDOS 386 - (ISA) DMA service
3293                              <1>
3294                              <1> end_of_syscalls:
3295                              <1>
3296                              <1> error:
3297                              <1>       ; 18/05/2016
3298                              <1>       ; 13/05/2016
3299                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3300                              <1>       ; 16/04/2015 - 17/09/2015 (Retro UNIX 386 v1)
3301                              <1>       ; 10/04/2013 - 07/08/2013 (Retro UNIX 8086 v1)
3302                              <1>       ;
3303                              <1>       ; 'error' merely sets the error bit off the processor status (c-bit)
3304                              <1>       ; then falls right into the 'sysret', 'sysrele' return sequence.
3305                              <1>       ;
3306                              <1>       ; INPUTS -> none
3307                              <1>       ; OUTPUTS ->
3308                              <1>       ;     processor status - carry (c) bit is set (means error)
3309                              <1>       ;
3310                              <1>       ; 26/05/2013 (Stack pointer must be reset here!
3311                              <1>       ;           Because, jumps to error procedure
3312                              <1>       ;           disrupts push-pop nesting balance)
3313                              <1>       ;
3314 0000D7AC 8B2D[5C030300]      <1>       mov    ebp, [u.sp] ; interrupt (system call) return (iretd) address
3315 0000D7B2 804D0801            <1>       or     byte [ebp+8], 1  ; set carry bit of flags register
3316                              <1>                               ; (system call will return with cf = 1)
3317                              <1>             ; bis $1,20.(r1) / set c bit in processor status word below
3318                              <1>                               ; / users stack
3319                              <1>       ; 17/09/2015
3320 0000D7B6 83ED30              <1>       sub    ebp, ESPACE ; 48 ; total size of stack frame ('sysdefs.inc')
3321                              <1>                              ; for saving/restoring user registers
3322                              <1>       ;cmp   ebp, [u.usp]
3323                              <1>       ;je    short err0
3324 0000D7B9 892D[60030300]      <1>       mov    [u.usp], ebp
3325                              <1> ;err0:
3326                              <1>       ; 01/09/2015
3327 0000D7BF 8B25[60030300]      <1>       mov    esp, [u.usp]     ; Retro Unix 8086 v1 modification!
3328                              <1>                               ; 10/04/2013
3329                              <1>                               ; (If an I/O error occurs during disk I/O,
3330                              <1>                               ; related procedures will jump to 'error'
3331                              <1>                               ; procedure directly without returning to
3332                              <1>                               ; the caller procedure. So, stack pointer
3333                              <1>                                     ; must be restored here.)
3334                              <1>       ; 13/05/2016
3335                              <1>       ; NOTE: (The last) error code is in 'u.error', it can be retrieved by
3336                              <1>       ;     'get last error' system call later.
3337                              <1>
3338                              <1>       ; 03/09/2015 - 09/06/2015 - 07/08/2013
3339 0000D7C5 C605[C6030300]00    <1>       mov    byte [u.kcall], 0 ; namei_r, mkdir_w reset
3340                              <1>
3341                              <1> sysret: ; < return from system call>
3342                              <1>       ; 01/03/2017
3343                              <1>       ; 28/02/2017
3344                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3345                              <1>       ; 16/04/2015 - 10/09/2015 (Retro UNIX 386 v1)
3346                              <1>       ; 10/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
3347                              <1>       ;
3348                              <1>       ; 'sysret' first checks to see if process is about to be
3349                              <1>       ; terminated (u.bsys). If it is, 'sysexit' is called.
3350                              <1>       ; If not, following happens:
3351                              <1>       ;     1) The user's stack pointer is restored.
3352                              <1>       ;     2) r1=0 and 'iget' is called to see if last mentioned
```

```
3353                                 <1>       ;         i-node has been modified. If it has, it is written out
3354                                 <1>       ;         via 'ppoke'.
3355                                 <1>       ;      3) If the super block has been modified, it is written out
3356                                 <1>       ;         via 'ppoke'.
3357                                 <1>       ;      4) If the dismountable file system's super block has been
3358                                 <1>       ;         modified, it is written out to the specified device
3359                                 <1>       ;         via 'ppoke'.
3360                                 <1>       ;      5) A check is made if user's time quantum (uquant) ran out
3361                                 <1>       ;         during his execution. If so, 'tswap' is called to give
3362                                 <1>       ;         another user a chance to run.
3363                                 <1>       ;      6) 'sysret' now goes into 'sysrele'.
3364                                 <1>       ;          (See 'sysrele' for conclusion.)
3365                                 <1>       ;
3366                                 <1>       ; Calling sequence:
3367                                 <1>       ;      jump table or 'br sysret'
3368                                 <1>       ; Arguments:
3369                                 <1>       ;      -
3370                                 <1>       ; ............................................................
3371                                 <1>       ;
3372                                 <1>       ; ((AX=r1 for 'iget' input))
3373                                 <1>       ;
3374 0000D7CC 31C0                  <1>       xor    eax, eax ; 28/02/2017
3375                                 <1> sysret0: ; 29/07/2015 (eax = 0, jump from sysexec)
3376 0000D7CE FEC0                  <1>       inc    al ; 04/05/2013
3377 0000D7D0 3805[B2030300]        <1>       cmp    [u.bsys], al ; 1
3378                                 <1>              ; tstb u.bsys / is a process about to be terminated because
3379 0000D7D6 0F8377010000          <1>       jnb    sysexit ; 04/05/2013
3380                                 <1>              ; bne sysexit / of an error? yes, go to sysexit
3381                                 <1>       ;mov   esp, [u.usp] ; 24/05/2013 (that is not needed here)
3382                                 <1>              ; mov u.sp,sp / no point stack to users stack
3383 0000D7DC FEC8                  <1>       dec    al ; mov ax, 0
3384                                 <1>              ; clr r1 / zero r1 to check last mentioned i-node
3385 0000D7DE E88A520000            <1>       call   iget
3386                                 <1>              ; jsr r0,iget / if last mentioned i-node has been modified
3387                                 <1>                              ; / it is written out
3388                                 <1>       ; 10/01/2017
3389                                 <1>       ; 09/01/2017
3390                                 <1> ;sysrele: ; < release >
3391                                 <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3392                                 <1>       ; 16/04/2015 - 14/10/2015 (Retro UNIX 386 v1)
3393                                 <1>       ; 10/04/2013 - 07/03/2014 (Retro UNIX 8086 v1)
3394                                 <1>       ;
3395                                 <1>       ; 'sysrele' first calls 'tswap' if the time quantum for a user is
3396                                 <1>       ;  zero (see 'sysret'). It then restores the user's registers and
3397                                 <1>       ; turns off the system flag. It then checked to see if there is
3398                                 <1>       ; an interrupt from the user by calling 'isintr'. If there is,
3399                                 <1>       ; the output gets flashed (see isintr) and interrupt action is
3400                                 <1>       ; taken by a branch to 'intract'. If there is no interrupt from
3401                                 <1>       ; the user, a rti is made.
3402                                 <1>       ;
3403                                 <1>       ; Calling sequence:
3404                                 <1>       ;      Fall through a 'bne' in 'sysret' & ?
3405                                 <1>       ; Arguments:
3406                                 <1>       ;      -
3407                                 <1>       ; ............................................................
3408                                 <1>       ;
3409                                 <1>       ; 23/02/2014 (swapret)
3410                                 <1>       ; 22/09/2013
3411                                 <1> sysrel0: ;1:
3412 0000D7E3 803D[A8030300]00      <1>       cmp    byte [u.quant], 0 ; 16/05/2013
3413                                 <1>              ; tstb uquant / is the time quantum 0?
3414 0000D7EA 7705                  <1>       ja     short swapret
3415                                 <1>              ; bne 1f / no, don't swap it out
3416                                 <1> sysrelease: ; 07/12/2013 (jump from 'clock')
3417 0000D7EC E83F400000            <1>       call   tswap
3418                                 <1>              ; jsr r0,tswap / yes, swap it out
3419                                 <1>
3420                                 <1> ; Retro Unix 8086 v1 feature: return from 'swap' to 'swapret' address.
3421                                 <1> swapret: ;1:
3422                                 <1>       ; 10/09/2015
3423                                 <1>       ; 01/09/2015
3424                                 <1>       ; 14/05/2015
3425                                 <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit, pm modifications)
3426                                 <1>       ; 26/05/2013 (Retro UNIX 8086 v1)
3427                                 <1>       ; cli
3428                                 <1>       ; 24/07/2015
3429                                 <1>       ;
3430                                 <1>       ;; 'esp' must be already equal to '[u.usp]' here !
3431                                 <1>       ;; mov esp, [u.usp]
3432                                 <1>
3433                                 <1>       ; 22/09/2013
3434 0000D7F1 E877520000            <1>       call   isintr
3435                                 <1>       ; 20/10/2013
3436 0000D7F6 7405                  <1>       jz     short sysrel1
3437 0000D7F8 E83F010000            <1>       call   intract
3438                                 <1>              ; jsr r0,isintr / is there an interrupt from the user
3439                                 <1>              ;     br intract / yes, output gets flushed, take interrupt
3440                                 <1>                                ; / action
3441                                 <1> sysrel1:
3442 0000D7FD FA                    <1>       cli    ; 14/10/2015
3443                                 <1> sysrel2:
3444                                 <1>       ; 28/02/2017
3445                                 <1>       ; Check if there is a (delayed) callback for current user/process
3446 0000D7FE A0[D7030300]          <1>       mov    al, [u.irqwait]
3447 0000D803 240F                  <1>       and    al, 0Fh ; is there a waiting IRQ callback service ?
3448 0000D805 7444                  <1>       jz     short sysrel8 ; no
3449                                 <1>
3450                                 <1>       ; Set return to IRQ callback service and return from the service
3451 0000D807 0FB6D8                <1>       movzx  ebx, al
3452 0000D80A 883D[D7030300]        <1>       mov    [u.irqwait], bh ; 0 ; reset
3453 0000D810 8A9B[B8480100]        <1>       mov    bl, [ebx+IRQenum] ; (available) IRQ index +1 (1 to 9)
3454                                 <1>       ; 01/03/2017
3455 0000D816 FECB                  <1>       dec    bl ; IRQ index number, 0 to 8
3456 0000D818 7831                  <1>       js     short sysrel8 ; 0 -> FFh (not in use!?)
3457                                 <1>       ;
```

```
3458 0000D81A A0[B3030300]        <1>       mov   al, [u.uno] ; current process (user) number
3459 0000D81F 3883[229B0100]      <1>       cmp   [ebx+IRQ.owner], al
3460 0000D825 7524                <1>       jne   short sysrel8 ; it is not the current user/process !?
3461 0000D827 F683[349B0100]01    <1>       test  byte [ebx+IRQ.method], 1 ; callback ?
3462 0000D82E 741B                <1>       jz    short sysrel8 ; not a callback method !?
3463                              <1>
3464 0000D830 8B93[469B0100]      <1>       mov   edx, [ebx+IRQ.addr] ; IRQ callback service address (virtual)
3465 0000D836 C605[D8030300]01    <1>       mov   byte [u.r_lock], 1 ; IRQ callback service in progress flag
3466                              <1>
3467 0000D83D E896400000          <1>       call  wswap ; save user's registers & status
3468                              <1>             ;      (for return from IRQ callback service)
3469                              <1>
3470 0000D842 8B2D[5C030300]      <1>       mov   ebp, [u.sp]; kernel's stack, points to EIP (user)
3471 0000D848 895500              <1>       mov   [ebp], edx ; IRQ call back service address
3472                              <1> sysrel8:
3473 0000D84B FE0D[5B030300]      <1>       dec   byte [sysflg]
3474                              <1>             ; decb sysflg / turn system flag off
3475                              <1>
3476 0000D851 A1[B8030300]        <1>       mov   eax, [u.pgdir]
3477 0000D856 0F22D8              <1>       mov   cr3, eax  ; 1st PDE points to Kernel Page Table 0 (1st 4 MB)
3478                              <1>                       ; (others are different than kernel page tables)
3479                              <1>       ; 10/09/2015
3480 0000D859 61                  <1>       popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
3481                              <1>             ; mov (sp)+,sc / restore user registers
3482                              <1>             ; mov (sp)+,mq
3483                              <1>             ; mov (sp)+,ac
3484                              <1>             ; mov (sp)+,r5
3485                              <1>             ; mov (sp)+,r4
3486                              <1>             ; mov (sp)+,r3
3487                              <1>             ; mov (sp)+,r2
3488                              <1>       ;
3489 0000D85A A1[64030300]        <1>       mov   eax, [u.r0]  ; ((return value in EAX))
3490 0000D85F 0FA9                <1>       pop   gs
3491 0000D861 0FA1                <1>       pop   fs
3492 0000D863 07                  <1>       pop   es
3493 0000D864 1F                  <1>       pop   ds
3494                              <1>       ;or   word [esp+8], 200h ; 22/01/2017 ; force enabling interrupts
3495 0000D865 CF                  <1>       iretd
3496                              <1>             ; rti / no, return from interrupt
3497                              <1>
3498                              <1> sysrele:
3499                              <1>       ; 24/03/2017
3500                              <1>       ; 28/02/2017
3501                              <1>       ; 27/02/2017
3502                              <1>       ; 29/01/2017
3503                              <1>       ; 14/01/2017
3504                              <1>       ; 13/01/2017
3505                              <1>       ; 09/01/2017, 10/01/2017, 12/01/2017
3506                              <1>       ; Major modification for TRDOS 386 (CallBack return)
3507                              <1>       ;
3508                              <1>       ; 'sysrele' system call restores previously saved
3509                              <1>       ; registers and addresses of the process
3510                              <1>       ; (Main purpose -in TRDOS 386- is to return from
3511                              <1>       ; timer callback service routine in ring 3 -user mode-.)
3512                              <1>       ;
3513                              <1>       ; check if the process is in timer callback phase
3514 0000D866 803D[D4030300]00    <1>       cmp   byte [u.t_lock], 0 ; TIMER INT LOCK
3515                              <1>       ;je   short sysrel0 ; classic (Retro UNIX 386 type) sysrele
3516 0000D86D 7734                <1>       ja    short sysrel3
3517                              <1>       ; 27/02/2017
3518 0000D86F 803D[D8030300]00    <1>       cmp   byte [u.r_lock], 0 ; IRQ callback lock
3519 0000D876 0F8667FFFFFF        <1>       jna   sysrel0 ; classic sysrele ; 24/03/2017
3520 0000D87C E859000000          <1>       call  sysrel7
3521 0000D881 803D[D8030300]00    <1>       cmp   byte [u.r_lock], 0 ; IRQ callback service lock
3522 0000D888 7628                <1>       jna   short sysrel4
3523 0000D88A C605[D8030300]00    <1>       mov   byte [u.r_lock], 0 ; reset
3524                              <1>       ;mov  byte [u.irqwait], 0 ; reset ; 28/02/2017
3525 0000D891 A0[D9030300]        <1>       mov   al, [u.r_mode]
3526 0000D896 08C0                <1>       or    al, al
3527 0000D898 7518                <1>       jnz   short sysrel4
3528 0000D89A FEC8                <1>       dec   al
3529 0000D89C A2[D9030300]        <1>       mov   [u.r_mode], al ; 0FFh ; not necessary !?
3530 0000D8A1 EB32                <1>       jmp   short sysrel6
3531                              <1> sysrel3:
3532                              <1>       ; 27/02/2017
3533 0000D8A3 E832000000          <1>       call  sysrel7
3534                              <1>       ; 14/01/2017
3535 0000D8A8 28C0                <1>       sub   al, al
3536 0000D8AA 3805[D4030300]      <1>       cmp   [u.t_lock], al ; 0 ; TIMER INT LOCK
3537 0000D8B0 770E                <1>       ja    short sysrel5 ; yes
3538                              <1> sysrel4:
3539                              <1>       ; 29/01/2017
3540 0000D8B2 8B44241C            <1>       mov   eax, [esp+28] ; eax
3541 0000D8B6 A3[64030300]        <1>       mov   [u.r0], eax
3542 0000D8BB E93EFFFFFF          <1>       jmp   sysrel2
3543                              <1> sysrel5:
3544 0000D8C0 A2[D4030300]        <1>       mov   [u.t_lock], al ; 0 ; reset
3545 0000D8C5 A0[D5030300]        <1>       mov   al, [u.t_mode]
3546 0000D8CA 20C0                <1>       and   al, al
3547                              <1>       ;jnz  short sysrel2 ; 0FFh ; user mode
3548 0000D8CC 75E4                <1>       jnz   short sysrel4 ; 29/01/2017
3549 0000D8CE FEC8                <1>       dec   al
3550 0000D8D0 A2[D5030300]        <1>       mov   [u.t_mode], al ; 0FFh ; not necessary !?
3551                              <1> sysrel6:
3552                              <1>       ; cpu will continue from the interrupted sytem call addr
3553 0000D8D5 61                  <1>       popad        ; edi, esi, ebp, esp, ebx, edx, ecx, eax
3554 0000D8D6 83C410              <1>       add   esp, 16      ; pass segment segisters: ds, es, fs, gs
3555 0000D8D9 CF                  <1>       iretd        ; eip, cs, eflags
3556                              <1>
3557                              <1> sysrel7:
3558 0000D8DA 0FB61D[B3030300]    <1>       movzx ebx, byte [u.uno] ; current process number
3559 0000D8E1 66C1E302            <1>       shl   bx, 2
3560                              <1>       ;cmp  [ebx+p.tcb-4], eax ; 0 ; is there callback address ?
3561                              <1>       ;jna  short sysrel0
3562                              <1>       ; yes, reset callback address then restore process registers
```

```
3563                                    <1>       ;mov   [ebx+p.tcb-4], eax ; 0 ; reset
3564 0000D8E5 8B83[BC000300]            <1>       mov     eax, [ebx+p.upage-4] ; UPAGE address
3565 0000D8EB FA                        <1>       cli    ; disable interrupts till 'iretd'
3566 0000D8EC E91F400000                <1>       jmp    rswap ; restore process 'u' structure
3567                                    <1>
3568                                    <1> badsys:
3569                                    <1>       ; 25/12/2016
3570                                    <1>       ; 18/04/2016 (TRDOS 386 = TRDOS v2.0)
3571                                    <1>       ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
3572                                    <1>       ; 03/02/2011 ('trdos_ifc_routine')
3573                                    <1>       ;
3574                                    <1>       ; 16/04/2015 (Retro UNIX 386 v1, 'badsys')
3575                                    <1>       ; (EIP, EAX values will be shown on screen with error message)
3576                                    <1>       ; (EIP = 'CD 40h' instruction address -INT 40h-)
3577                                    <1>       ; (EAX = Function number)
3578                                    <1>       ;
3579 0000D8F1 FE05[B2030300]            <1>       inc    byte [u.bsys]
3580                                    <1>       ;
3581 0000D8F7 8B1D[5C030300]            <1>       mov    ebx, [u.sp] ; esp at the beginning of 'sysent'
3582 0000D8FD 8B03                      <1>       mov    eax, [ebx] ; EIP (return address, not 'INT 30h' address)
3583 0000D8FF 83E802                    <1>       sub    eax, 2 ; CDh, ##h
3584 0000D902 E86969FFFF                <1>       call   dwordtohex
3585 0000D907 8915[91460100]            <1>       mov    [eip_str], edx
3586 0000D90D A3[95460100]              <1>       mov    [eip_str+4], eax
3587 0000D912 A1[64030300]              <1>       mov    eax, [u.r0]
3588 0000D917 E85469FFFF                <1>       call   dwordtohex
3589 0000D91C 8915[80460100]            <1>       mov    [eax_str], edx
3590 0000D922 A3[84460100]              <1>       mov    [eax_str+4], eax
3591                                    <1>
3592 0000D927 66C705[75460100]34-       <1>       mov    word [int_num_str], SYSCALL_INT_NUM ; 25/12/2016
3592 0000D92F 30                        <1>
3593                                    <1>
3594 0000D930 BE[47460100]              <1>       mov    esi, ifc_msg ; "invalid funtion call !" msg (trdosk9.s)
3595 0000D935 E8EB9AFFFF                <1>       call   print_msg
3596                                    <1>
3597 0000D93A EB17                      <1>       jmp    sysexit
3598                                    <1>
3599                                    <1> intract: ; / interrupt action
3600                                    <1>       ; 14/10/2015
3601                                    <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
3602                                    <1>       ; 09/05/2013 - 07/12/2013 (Retro UNIX 8086 v1)
3603                                    <1>       ;
3604                                    <1>       ; Retro UNIX 8086 v1 modification !
3605                                    <1>       ; (Process/task switching and quit routine by using
3606                                    <1>       ; Retro UNIX 8086 v1 keyboard interrupt output.))
3607                                    <1>       ;
3608                                    <1>       ; input -> 'u.quit'  (also value of 'u.intr' > 0)
3609                                    <1>       ; output -> If value of 'u.quit' = FFFFh ('ctrl+brk' sign)
3610                                    <1>       ;            'intract' will jump to 'sysexit'.
3611                                    <1>       ;          Intract will return to the caller
3612                                    <1>       ;            if value of 'u.quit' <> FFFFh.
3613                                    <1>       ; 14/10/2015
3614 0000D93C FB                        <1>       sti
3615                                    <1>       ; 07/12/2013
3616 0000D93D 66FF05[AC030300]          <1>       inc    word [u.quit]
3617 0000D944 7408                      <1>       jz     short intrct0 ; FFFFh -> 0
3618 0000D946 66FF0D[AC030300]          <1>       dec    word [u.quit]
3619                                    <1>       ; 16/04/2015
3620 0000D94D C3                        <1>       retn
3621                                    <1> intrct0:
3622 0000D94E 58                        <1>       pop    eax ; call intract -> retn
3623                                    <1>       ;
3624 0000D94F 31C0                      <1>       xor    eax, eax
3625 0000D951 FEC0                      <1>       inc    al  ; mov ax, 1
3626                                    <1> ;;;
3627                                    <1>       ; UNIX v1 original 'intract' routine...
3628                                    <1>       ; / interrupt action
3629                                    <1>             ;cmp *(sp),$rti / are you in a clock interrupt?
3630                                    <1>             ; bne 1f / no, 1f
3631                                    <1>             ; cmp (sp)+,(sp)+ / pop clock pointer
3632                                    <1>       ; 1: / now in user area
3633                                    <1>             ; mov r1,-(sp) / save r1
3634                                    <1>             ; mov u.ttyp,r1
3635                                    <1>                   ; / pointer to tty buffer in control-to r1
3636                                    <1>             ; cmpb 6(r1),$177
3637                                    <1>                   ; / is the interrupt char equal to "del"
3638                                    <1>             ; beq 1f / yes, 1f
3639                                    <1>             ; clrb 6(r1)
3640                                    <1>                    ; / no, clear the byte
3641                                    <1>                   ; / (must be a quit character)
3642                                    <1>             ; mov (sp)+,r1 / restore r1
3643                                    <1>             ; clr u.quit / clear quit flag
3644                                    <1>             ; bis $20,2(sp)
3645                                    <1>                   ; / set trace for quit (sets t bit of
3646                                    <1>                   ; / ps-trace trap)
3647                                    <1>             ; rti   ;  / return from interrupt
3648                                    <1>       ; 1: / interrupt char = del
3649                                    <1>             ; clrb 6(r1) / clear the interrupt byte
3650                                    <1>                      ; / in the buffer
3651                                    <1>             ; mov (sp)+,r1 / restore r1
3652                                    <1>             ; cmp u.intr,$core / should control be
3653                                    <1>                         ; / transferred to loc core?
3654                                    <1>             ; blo 1f
3655                                    <1>             ; jmp *u.intr / user to do rti yes,
3656                                    <1>                         ; / transfer to loc core
3657                                    <1>       ; 1:
3658                                    <1>             ; sys 1 / exit
3659                                    <1>
3660                                    <1> sysexit: ; <terminate process>
3661                                    <1>       ; 14/11/2017
3662                                    <1>       ; 27/05/2017
3663                                    <1>       ; 10/04/2017
3664                                    <1>       ; 26/02/2017, 28/02/2017
3665                                    <1>       ; 02/01/2017, 23/01/2017
3666                                    <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
0000D953 6648                   <1>       dec   ax ; 0
0000D955 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
0000D95B FA                     <1>       cli   ; disable interrupts
                                <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
0000D95C B036                   <1>       mov   al, 00110110b ; 36h
0000D95E E643                   <1>       out   43h, al
0000D960 28C0                   <1>       sub   al, al ; 0
0000D962 E640                   <1>       out   40h, al ; LB
0000D964 E640                   <1>       out   40h, al ; HB
                                <1>       ;
0000D966 0FB61D[B3030300]       <1>       movzx ebx, byte [u.uno]
                                <1>       ;mov  bl, [u.uno] ; process number of dying process
0000D96D 3883[FF000300]         <1>       cmp   byte [ebx+p.timer-1], al ; 0
0000D973 763A                   <1>       jna   short sysexit_12 ; no timer events for this process
0000D975 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
0000D97B 8A0D[83950100]         <1>       mov   cl, [timer_events] ; 14/11/2017
                                <1>       ;cli  ; disable interrupts
0000D981 B410                   <1>       mov   ah, 16 ; number of available timer events
0000D983 BE[60040300]           <1>       mov   esi, timer_set ; beginning address of timer events
                                <1> sysexit_7:
0000D988 8A06                   <1>       mov   al, [esi] ; process number (of timer event)
0000D98A 38D8                   <1>       cmp   al, bl ; process number comparison
0000D98C 7411                   <1>       je    short sysexit_10
0000D98E 20C0                   <1>       and   al, al
0000D990 7404                   <1>       jz    short sysexit_9
                                <1> sysexit_8:
0000D992 FEC9                   <1>       dec   cl
0000D994 7416                   <1>       jz    short sysexit_11
                                <1> sysexit_9:
0000D996 FECC                   <1>       dec   ah
0000D998 7415                   <1>       jz    short sysexit_12
0000D99A 83C610                 <1>       add   esi, 16
0000D99D EBE9                   <1>       jmp   short sysexit_7
                                <1>
                                <1> sysexit_10:
                                <1>       ;mov  byte [esi], 0
0000D99F 66C7060000             <1>       mov   word [esi], 0
                                <1>       ;mov  dword [esi+12], 0
                                <1>       ;
0000D9A4 FE0D[83950100]         <1>       dec   byte [timer_events] ; 02/01/2017
                                <1>       ;
0000D9AA EBE6                   <1>       jmp   short sysexit_8
                                <1>
                                <1> sysexit_11:
0000D9AC 6629C0                 <1>       sub   ax, ax ; 0 ; 26/02/2017
                                <1> sysexit_12:
```

```
3772                                    <1>       ; 26/02/2017 (Unlink IRQ callbacks belong to the user)
3773 0000D9AF 803D[D6030300]00          <1>       cmp    byte [u.irqc], 0 ; Count of IRQ callbacks
3774 0000D9B6 7E2E                      <1>       jng    short sysexit_16 ; zero or invalid
3775                                    <1>       ; 28/02/2017
3776                                    <1>       ; clear IRQ callback flags (for 'sysrele' and 'sysret')
3777 0000D9B8 A2[D7030300]              <1>       mov    [u.irqwait], al ; 0 ; force to clear waiting flag
3778 0000D9BD A2[D8030300]              <1>       mov    [u.r_lock], al ; 0 ; force to clear busy flag
3779 0000D9C2 BE[229B0100]              <1>       mov    esi, IRQ.owner
3780                                    <1> sysexit_13:
3781 0000D9C7 AC                        <1>       lodsb
3782 0000D9C8 3A05[B3030300]            <1>       cmp    al, [u.uno] ; owner = current user ?
3783 0000D9CE 750C                      <1>       jne    short sysexit_14
3784 0000D9D0 C646FF00                  <1>       mov    byte [esi-1], 0 ; owner = 0 : Free
3785 0000D9D4 FE0D[D6030300]            <1>       dec    byte [u.irqc]
3786 0000D9DA 7408                      <1>       jz     short sysexit_15
3787                                    <1> sysexit_14:
3788 0000D9DC 81FE[2A9B0100]            <1>       cmp    esi, IRQ.owner + 8 ; the last IRQ index number ?
3789 0000D9E2 76E3                      <1>       jna    short sysexit_13 ; no
3790                                    <1> sysexit_15:
3791 0000D9E4 30C0                      <1>       xor    al, al ; 0
3792                                    <1> sysexit_16: ; 2:
3793 0000D9E6 FB                        <1>       sti    ; enable interrupts
3794                                    <1>       ;
3795                                    <1>       ; AX = 0
3796                                    <1> sysexit_1: ; 1:
3797                                    <1>       ; AX = File descriptor
3798                                    <1>             ; / r1 has file descriptor (index to u.fp list)
3799                                    <1>             ; / Search the whole list
3800 0000D9E7 E811350000                <1>       call   fclose
3801                                    <1>             ; jsr r0,fclose / close all files the process opened
3802                                    <1>       ;; ignore error return
3803                                    <1>             ; br .+2 / ignore error return
3804                                    <1>       ;inc   ax
3805 0000D9EC FEC0                      <1>       inc    al
3806                                    <1>             ; inc r1 / increment file descriptor
3807                                    <1>       ;cmp   ax, 10
3808 0000D9EE 3C0A                      <1>       cmp    al, 10
3809                                    <1>             ; cmp r1,$10. / end of u.fp list?
3810 0000D9F0 72F5                      <1>       jb     short sysexit_1
3811                                    <1>             ; blt 1b / no, go back
3812                                    <1>       ;movzx ebx, byte [u.uno]
3813 0000D9F2 8A1D[B3030300]            <1>       mov    bl, [u.uno] ; 02/01/2017
3814                                    <1>             ; movb u.uno,r1 / yes, move dying process's number to r1
3815 0000D9F8 88A3[AF000300]            <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
3816                                    <1>             ; clrb p.stat-1(r1) / free the process
3817                                    <1>       ; 10/04/2017
3818 0000D9FE 381D[999B0100]            <1>       cmp    [audio_user], bl
3819 0000DA04 7518                      <1>       jne    short sysexit_17
3820                                    <1>       ; reset audio device (current) owner and 'initializated' flag
3821 0000DA06 883D[999B0100]            <1>       mov    [audio_user], bh ; 0
3822                                    <1>       ; 27/05/2017
3823 0000DA0C 8B0D[849B0100]            <1>       mov    ecx,  [audio_buffer]
3824 0000DA12 09C9                      <1>       or     ecx, ecx
3825 0000DA14 7408                      <1>       jz     short sysexit_17
3826                                    <1>       ; 'deallocate_user_pages' is not necessary in sysexit !!!
3827                                    <1>       ;push  ebx
3828                                    <1>       ;mov   ebx, ecx
3829                                    <1>       ;mov   ecx, [audio_buff_size]
3830                                    <1>       ;call  deallocate_user_pages
3831                                    <1>       ;; (Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP)
3832 0000DA16 29C9                      <1>       sub    ecx, ecx
3833 0000DA18 890D[849B0100]            <1>       mov    [audio_buffer], ecx ; 0
3834                                    <1>       ;pop   ebx
3835                                    <1> sysexit_17:
3836                                    <1>       ;shl   bx, 1
3837 0000DA1E D0E3                      <1>       shl    bl, 1
3838                                    <1>             ; asl r1 / use r1 for index into the below tables
3839 0000DA20 668B8B[1E000300]          <1>       mov    cx, [ebx+p.pid-2]
3840                                    <1>             ; mov p.pid-2(r1),r3 / move dying process's name to r3
3841 0000DA27 668B93[3E000300]          <1>       mov    dx, [ebx+p.ppid-2]
3842                                    <1>             ; mov p.ppid-2(r1),r4 / move its parents name to r4
3843                                    <1>       ; xor  bx, bx ; 0
3844 0000DA2E 30DB                      <1>       xor    bl, bl ; 0
3845                                    <1>             ; clr r2
3846 0000DA30 31F6                      <1>       xor    esi, esi ; 0
3847                                    <1>             ; clr r5 / initialize reg
3848                                    <1> sysexit_2: ; 1:
3849                                    <1>              ; / find children of this dying process,
3850                                    <1>             ; / if they are zombies, free them
3851                                    <1>       ;add   bx, 2
3852 0000DA32 80C302                    <1>       add    bl, 2
3853                                    <1>             ; add $2,r2 / search parent process table
3854                                    <1>                       ; / for dying process's name
3855 0000DA35 66398B[3E000300]          <1>       cmp    [ebx+p.ppid-2], cx
3856                                    <1>             ; cmp p.ppid-2(r2),r3 / found it?
3857 0000DA3C 7513                      <1>       jne    short sysexit_4
3858                                    <1>             ; bne 3f / no
3859                                    <1>       ;shr   bx, 1
3860 0000DA3E D0EB                      <1>       shr    bl, 1
3861                                    <1>             ; asr r2 / yes, it is a parent
3862 0000DA40 80BB[AF000300]03          <1>       cmp    byte [ebx+p.stat-1], 3 ; SZOMB
3863                                    <1>             ; cmpb p.stat-1(r2),$3 / is the child of this
3864                                    <1>                                    ; / dying process a zombie
3865 0000DA47 7506                      <1>       jne    short sysexit_3
3866                                    <1>             ; bne 2f / no
3867 0000DA49 88A3[AF000300]            <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
3868                                    <1>             ; clrb p.stat-1(r2) / yes, free the child process
3869                                    <1> sysexit_3: ; 2:
3870                                    <1>       ;shr   bx, 1
3871 0000DA4F D0E3                      <1>       shl    bl, 1
3872                                    <1>             ; asl r2
3873                                    <1> sysexit_4: ; 3:
3874                                    <1>             ; / search the process name table
3875                                    <1>             ; / for the dying process's parent
3876 0000DA51 663993[1E000300]          <1>       cmp    [ebx+p.pid-2], dx
```

```
                                <1>                     ; cmp p.pid-2(r2),r4 / found it?
0000DA58 7502                   <1>             jne     short sysexit_5
                                <1>                     ; bne 3f / no
0000DA5A 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
0000DA5C 80FB20                 <1>             cmp     bl, nproc + nproc
                                <1>                     ; cmp r2,$nproc+nproc / has whole table been searched?
0000DA5F 72D1                   <1>             jb      short sysexit_2
                                <1>                     ; blt 1b / no, go back
                                <1>                     ; mov r5,r1 / yes, r1 now has parents process # x2
0000DA61 21F6                   <1>             and     esi, esi ; r5=r1
0000DA63 7436                   <1>             jz      short sysexit_6
                                <1>                     ; beq 2f / no parent has been found.
                                <1>                           ; / The process just dies
0000DA65 66D1EE                 <1>             shr     si, 1
                                <1>                     ; asr r1 / set up index to p.stat
0000DA68 8A86[AF000300]         <1>             mov     al, [esi+p.stat-1]
                                <1>                     ; movb p.stat-1(r1),r2 / move status of parent to r2
0000DA6E 20C0                   <1>             and     al, al
0000DA70 7429                   <1>             jz      short sysexit_6
                                <1>                     ; beq 2f / if its been freed, 2f
0000DA72 3C03                   <1>             cmp     al, 3
                                <1>                     ; cmp r2,$3 / is parent a zombie?
0000DA74 7425                   <1>             je      short sysexit_6
                                <1>                     ; beq 2f / yes, 2f
                                <1>             ; BH = 0
0000DA76 8A1D[B3030300]         <1>             mov     bl, [u.uno]
                                <1>                     ; movb u.uno,r3 / move dying process's number to r3
0000DA7C C683[AF000300]03       <1>             mov     byte [ebx+p.stat-1], 3  ; SZOMB
                                <1>                     ; movb $3,p.stat-1(r3) / make the process a zombie
0000DA83 3C01                   <1>             cmp     al, 1 ; SRUN
0000DA85 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
0000DA87 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)
0000DA8E 6689F0                 <1>             mov     ax, si ; r1  (process number in AL)
                                <1>             ;
                                <1>             ;mov    ebx, runq + 4
                                <1>                     ; mov $runq+4,r2 / on the runq
0000DA91 BB[54030300]           <1>             mov     ebx, runq+2 ; normal run queue ; 02/01/2017
0000DA96 E8AD3E0000             <1>             call    putlu
                                <1>                     ; jsr r0, putlu
                                <1> sysexit_6:
                                <1>                     ; / the process dies
0000DA9B C605[B3030300]00       <1>             mov     byte [u.uno], 0
                                <1>                     ; clrb u.uno / put zero as the process number,
                                <1>                         ; / so "swap" will
0000DAA2 E8A33D0000             <1>             call    swap
                                <1>                     ; jsr r0,swap / overwrite process with another process
                                <1> hlt_sys:
                                <1>             ;sti
                                <1> hlts0:
0000DAA7 F4                     <1>             hlt
0000DAA8 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>             ;
                                <1>
                                <1> ; / wait for a process to die
                                <1>
                                <1> syswait_0:
```

```
3982 0000DAAA 0FB61D[B3030300]    <1>       movzx  ebx, byte [u.uno] ; 01/09/2015
3983                              <1>              ; movb u.uno,r1 / put parents process number in r1
3984 0000DAB1 D0E3                <1>       shl    bl, 1
3985                              <1>       ;shl   bx, 1
3986                              <1>              ; asl r1 / x2 to get index into p.pid table
3987 0000DAB3 668B83[1E000300]    <1>       mov    ax, [ebx+p.pid-2]
3988                              <1>              ; mov p.pid-2(r1),r1 / get the name of this process
3989 0000DABA 31F6                <1>       xor    esi, esi
3990                              <1>              ; clr r2
3991 0000DABC 31C9                <1>       xor    ecx, ecx ; 30/10/2013
3992                              <1>       ;xor   cl, cl
3993                              <1>              ; clr r3 / initialize reg 3
3994                              <1> syswait_1: ; 1:
3995 0000DABE 6683C602            <1>       add    si, 2
3996                              <1>              ; add $2,r2 / use r2 for index into p.ppid table
3997                              <1>                      ; / search table of parent processes
3998                              <1>                      ; / for this process name
3999 0000DAC2 663B86[3E000300]    <1>       cmp    ax, [esi+p.ppid-2]
4000                              <1>              ; cmp p.ppid-2(r2),r1 / r2 will contain the childs
4001                              <1>                                   ; / process number
4002 0000DAC9 7535                <1>       jne    short syswait_3
4003                              <1>              ;bne 3f / branch if no match of parent process name
4004                              <1>       ;inc   cx
4005 0000DACB FEC1                <1>       inc    cl
4006                              <1>              ;inc r3 / yes, a match, r3 indicates number of children
4007 0000DACD 66D1EE              <1>       shr    si, 1
4008                              <1>              ; asr r2 / r2/2 to get index to p.stat table
4009                              <1>       ; The possible states ('p.stat' values) of a process are:
4010                              <1>       ;      0 = free or unused
4011                              <1>       ;      1 = active
4012                              <1>       ;      2 = waiting for a child process to die
4013                              <1>       ;      3 = terminated, but not yet waited for (zombie).
4014 0000DAD0 80BE[AF000300]03    <1>       cmp    byte [esi+p.stat-1], 3 ; SZOMB, 05/02/2014
4015                              <1>              ; cmpb p.stat-1(r2),$3 / is the child process a zombie?
4016 0000DAD7 7524                <1>       jne    short syswait_2
4017                              <1>              ; bne 2f / no, skip it
4018 0000DAD9 88BE[AF000300]      <1>       mov    [esi+p.stat-1], bh ; 0
4019                              <1>              ; clrb p.stat-1(r2) / yes, free it
4020 0000DADF 66D1E6              <1>       shl    si, 1
4021                              <1>              ; asl r2 / r2x2 to get index into p.pid table
4022 0000DAE2 0FB786[1E000300]    <1>       movzx  eax, word [esi+p.pid-2]
4023 0000DAE9 A3[64030300]        <1>       mov    [u.r0], eax
4024                              <1>              ; mov p.pid-2(r2),*u.r0
4025                              <1>                           ; / put childs process name in (u.r0)
4026                              <1>       ;
4027                              <1>       ; Retro UNIX 386 v1 modification ! (17/09/2015)
4028                              <1>       ;
4029                              <1>       ; Parent process ID -p.ppid- field (of the child process)
4030                              <1>       ; must be cleared in order to prevent infinitive 'syswait'
4031                              <1>       ; system call loop from the application/program if it calls
4032                              <1>       ; 'syswait' again (mistakenly) while there is not a zombie
4033                              <1>       ; or running child process to wait. ('forktest.s', 17/09/2015)
4034                              <1>       ;
4035                              <1>       ; Note: syswait will return with error if there is not a
4036                              <1>       ;       zombie or running process to wait.
4037                              <1>       ;
4038 0000DAEE 6629C0              <1>       sub    ax, ax
4039 0000DAF1 668986[3E000300]    <1>       mov    [esi+p.ppid-2], ax ; 0 ; 17/09/2015
4040 0000DAF8 E9D1FCFFFF          <1>       jmp    sysret0 ; ax = 0
4041                              <1>       ;
4042                              <1>       ;jmp   sysret
4043                              <1>              ; br sysret1 / return cause child is dead
4044                              <1> syswait_2: ; 2:
4045 0000DAFD 66D1E6              <1>       shl    si, 1
4046                              <1>              ; asl r2 / r2x2 to get index into p.ppid table
4047                              <1> syswait_3: ; 3:
4048 0000DB00 6683FE20            <1>       cmp    si, nproc+nproc
4049                              <1>              ; cmp r2,$nproc+nproc / have all processes been checked?
4050 0000DB04 72B8                <1>       jb     short syswait_1
4051                              <1>              ; blt 1b / no, continue search
4052                              <1>       ;and   cx, cx
4053 0000DB06 20C9                <1>       and    cl, cl
4054                              <1>              ; tst r3 / one gets here if there are no children
4055                              <1>                     ; / or children that are still active
4056                              <1>       ; 30/10/2013
4057 0000DB08 750B                <1>       jnz    short syswait_4
4058                              <1>       ;jz    error
4059                              <1>              ; beq error1 / there are no children, error
4060 0000DB0A 890D[64030300]      <1>       mov    [u.r0], ecx ; 0
4061 0000DB10 E997FCFFFF          <1>       jmp    error
4062                              <1> syswait_4:
4063 0000DB15 8A1D[B3030300]      <1>       mov    bl, [u.uno]
4064                              <1>              ; movb u.uno,r1 / there are children so put
4065                              <1>                           ; / parent process number in r1
4066 0000DB1B FE83[AF000300]      <1>       inc    byte [ebx+p.stat-1] ; 2, SWAIT, 05/02/2014
4067                              <1>              ; incb p.stat-1(r1) / it is waiting for
4068                              <1>                              ; / other children to die
4069                              <1>       ; 04/11/2013
4070 0000DB21 E8243D0000          <1>       call   swap
4071                              <1>              ; jsr r0,swap / swap it out, because it's waiting
4072 0000DB26 EB82                <1>       jmp    syswait_0
4073                              <1>              ; br syswait / wait on next process
4074                              <1>
4075                              <1> sysfork: ; < create a new process >
4076                              <1>       ; 02/01/2017 (TRDOS 386 modification)
4077                              <1>       ; 04/09/2015, 18/05/2015
4078                              <1>       ; 28/08/2015, 01/09/2015, 02/09/2015
4079                              <1>       ; 09/05/2015, 10/05/2015, 14/05/2015
4080                              <1>       ; 06/05/2015 (Retro UNIX 386 v1 - Beginning)
4081                              <1>       ; 24/05/2013 - 14/02/2014 (Retro UNIX 8086 v1)
4082                              <1>       ;
4083                              <1>       ; 'sysfork' creates a new process. This process is referred
4084                              <1>       ; to as the child process. This new process core image is
4085                              <1>       ; a copy of that of the caller of 'sysfork'. The only
4086                              <1>       ; distinction is the return location and the fact that (u.r0)
```

```
4087                    <1>     ; in the old process (parent) contains the process id (p.pid)
4088                    <1>     ; of the new process (child). This id is used by 'syswait'.
4089                    <1>     ; 'sysfork' works in the following manner:
4090                    <1>     ;    1) The process status table (p.stat) is searched to find
4091                    <1>     ;      a process number that is unused. If none are found
4092                    <1>     ;      an error occurs.
4093                    <1>     ;    2) when one is found, it becomes the child process number
4094                    <1>     ;      and it's status (p.stat) is set to active.
4095                    <1>     ;    3) If the parent had a control tty, the interrupt
4096                    <1>     ;      character in that tty buffer is cleared.
4097                    <1>     ;    4) The child process is put on the lowest priority run
4098                    <1>     ;      queue via 'putlu'.
4099                    <1>     ;    5) A new process name is gotten from 'mpid' (actually
4100                    <1>     ;      it is a unique number) and is put in the child's unique
4101                    <1>     ;      identifier; process id (p.pid).
4102                    <1>     ;    6) The process name of the parent is then obtained and
4103                    <1>     ;      placed in the unique identifier of the parent process
4104                    <1>     ;      name is then put in 'u.r0'.
4105                    <1>     ;    7) The child process is then written out on disk by
4106                    <1>     ;      'wswap',i.e., the parent process is copied onto disk
4107                    <1>     ;      and the child is born. (The child process is written
4108                    <1>     ;      out on disk/drum with 'u.uno' being the child process
4109                    <1>     ;      number.)
4110                    <1>     ;    8) The parent process number is then restored to 'u.uno'.
4111                    <1>     ;    9) The child process name is put in 'u.r0'.
4112                    <1>     ;   10) The pc on the stack sp + 18 is incremented by 2 to
4113                    <1>     ;      create the return address for the parent process.
4114                    <1>     ;   11) The 'u.fp' list as then searched to see what files
4115                    <1>     ;      the parent has opened. For each file the parent has
4116                    <1>     ;      opened, the corresponding 'fsp' entry must be updated
4117                    <1>     ;      to indicate that the child process also has opened
4118                    <1>     ;      the file. A branch to 'sysret' is then made.
4119                    <1>     ;
4120                    <1>     ; Calling sequence:
4121                    <1>     ;      from shell ?
4122                    <1>     ; Arguments:
4123                    <1>     ;      -
4124                    <1>     ; Inputs: -
4125                    <1>     ; Outputs: *u.r0 - child process name
4126                    <1>     ; ............................................................
4127                    <1>     ;
4128                    <1>     ; Retro UNIX 8086 v1 modification:
4129                    <1>     ;      AX = r0 = PID (>0) (at the return of 'sysfork')
4130                    <1>     ;      = process id of child a parent process returns
4131                    <1>     ;      = process id of parent when a child process returns
4132                    <1>     ;
4133                    <1>     ;       In original UNIX v1, sysfork is called and returns as
4134                    <1>     ;      in following manner: (with an example: c library, fork)
4135                    <1>     ;
4136                    <1>     ;      1:
4137                    <1>     ;            sys    fork
4138                    <1>     ;                   br 1f  / child process returns here
4139                    <1>     ;            bes    2f     / parent process returns here
4140                    <1>     ;            / pid of new process in r0
4141                    <1>     ;            rts    pc
4142                    <1>     ;      2: / parent process condionally branches here
4143                    <1>     ;            mov    $-1,r0 / pid = -1 means error return
4144                    <1>     ;            rts    pc
4145                    <1>     ;
4146                    <1>     ;      1: / child process brances here
4147                    <1>     ;            clr    r0   / pid = 0 in child process
4148                    <1>     ;            rts    pc
4149                    <1>     ;
4150                    <1>     ;      In UNIX v7x86 (386) by Robert Nordier (1999)
4151                    <1>     ;            // pid = fork();
4152                    <1>     ;            //
4153                    <1>     ;            // pid == 0 in child process;
4154                    <1>     ;            // pid == -1 means error return
4155                    <1>     ;            // in child,
4156                    <1>     ;            //     parents id is in par_uid if needed
4157                    <1>     ;
4158                    <1>     ;            _fork:
4159                    <1>     ;                  mov    $.fork,eax
4160                    <1>     ;                  int    $0x30
4161                    <1>     ;                  jmp    1f
4162                    <1>     ;                  jnc    2f
4163                    <1>     ;                  jmp    cerror
4164                    <1>     ;            1:
4165                    <1>     ;                  mov    eax,_par_uid
4166                    <1>     ;                  xor    eax,eax
4167                    <1>     ;            2:
4168                    <1>     ;                  ret
4169                    <1>     ;
4170                    <1>     ;      In Retro UNIX 8086 v1,
4171                    <1>     ;      'sysfork' returns in following manner:
4172                    <1>     ;
4173                    <1>     ;            mov    ax, sys_fork
4174                    <1>     ;            mov    bx, offset @f ; routine for child
4175                    <1>     ;            int    20h
4176                    <1>     ;            jc     error
4177                    <1>     ;
4178                    <1>     ;      ; Routine for parent process here (just after 'jc')
4179                    <1>     ;            mov    word ptr [pid_of_child], ax
4180                    <1>     ;            jmp    next_routine_for_parent
4181                    <1>     ;
4182                    <1>     ;      @@: ; routine for child process here
4183                    <1>     ;           ....
4184                    <1>     ;      NOTE: 'sysfork' returns to specified offset
4185                    <1>     ;            for child process by using BX input.
4186                    <1>     ;           (at first, parent process will return then
4187                    <1>     ;           child process will return -after swapped in-
4188                    <1>     ;           'syswait' is needed in parent process
4189                    <1>     ;           if return from child process will be waited for.)
4190                    <1>     ;
```

```
4191                            <1>
4192                            <1> ; / create a new process
4193                            <1>       ; EBX = return address for child process
4194                            <1>            ; (Retro UNIX 8086 v1 modification !)
4195 0000DB28 31F6               <1>       xor    esi, esi
4196                            <1>              ; clr r1
4197                            <1> sysfork_1: ; 1: / search p.stat table for unused process number
4198 0000DB2A 46                 <1>       inc    esi
4199                            <1>              ; inc r1
4200 0000DB2B 80BE[AF000300]00   <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE, 05/02/2014
4201                            <1>              ; tstb p.stat-1(r1) / is process active, unused, dead
4202 0000DB32 760B               <1>       jna    short sysfork_2
4203                            <1>              ; beq 1f / it's unused so branch
4204 0000DB34 6683FE10           <1>       cmp    si, nproc
4205                            <1>              ; cmp r1,$nproc / all processes checked
4206 0000DB38 72F0               <1>       jb     short sysfork_1
4207                            <1>              ; blt 1b / no, branch back
4208                            <1>       ;
4209                            <1>       ; Retro UNIX 8086 v1. modification:
4210                            <1>       ;      Parent process returns from 'sysfork' to address
4211                            <1>       ;      which is just after 'sysfork' system call in parent
4212                            <1>       ;      process. Child process returns to address which is put
4213                            <1>       ;      in BX register by parent process for 'sysfork'.
4214                            <1>       ;
4215                            <1>              ;add $2,18.(sp) / add 2 to pc when trap occured, points
4216                            <1>                              ; / to old process return
4217                            <1>              ; br error1 / no room for a new process
4218 0000DB3A E96DFCFFFF         <1>       jmp    error
4219                            <1> sysfork_2: ; 1:
4220 0000DB3F E8637FFFFF         <1>       call   allocate_page
4221 0000DB44 0F8262FCFFFF       <1>       jc     error
4222 0000DB4A 50                 <1>       push   eax   ; UPAGE (user structure page) address
4223                            <1>       ; Retro UNIX 386 v1 modification!
4224 0000DB4B E86681FFFF         <1>       call   duplicate_page_dir
4225                            <1>              ; EAX = New page directory
4226 0000DB50 730B               <1>       jnc    short sysfork_3
4227 0000DB52 58                 <1>       pop    eax   ; UPAGE (user structure page) address
4228 0000DB53 E82D81FFFF         <1>       call   deallocate_page
4229 0000DB58 E94FFCFFFF         <1>       jmp    error
4230                            <1> sysfork_3:
4231                            <1>       ; Retro UNIX 386 v1 modification !
4232 0000DB5D 56                 <1>       push   esi
4233 0000DB5E E8753D0000         <1>       call   wswap ; save current user (u) structure, user registers
4234                            <1>                    ; and interrupt return components (for IRET)
4235 0000DB63 8705[B8030300]     <1>       xchg   eax, [u.pgdir] ; page directory of the child process
4236 0000DB69 A3[BC030300]       <1>       mov    [u.ppgdir], eax ; page directory of the parent process
4237 0000DB6E 5E                 <1>       pop    esi
4238 0000DB6F 58                 <1>       pop    eax   ; UPAGE (user structure page) address
4239                            <1>              ; [u.usp] = esp
4240 0000DB70 89F7               <1>       mov    edi, esi
4241 0000DB72 66C1E702           <1>       shl    di, 2
4242 0000DB76 8987[BC000300]     <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
4243 0000DB7C A3[B4030300]       <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
4244                            <1>       ; 28/08/2015
4245 0000DB81 0FB605[B3030300]   <1>       movzx  eax, byte [u.uno] ; parent process number
4246                            <1>              ; movb u.uno,-(sp) / save parent process number
4247 0000DB88 89C7               <1>       mov    edi, eax
4248 0000DB8A 50                 <1>         push eax ; **
4249 0000DB8B 8A87[7F000300]     <1>       mov     al, [edi+p.ttyc-1] ; console tty (parent)
4250                            <1>       ; 18/09/2015
4251                            <1>       ;mov     [esi+p.ttyc-1], al ; set child's console tty
4252                            <1>       ;mov     [esi+p.waitc-1], ah ; 0 ; reset child's wait channel
4253 0000DB91 668986[7F000300]   <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
4254                            <1>                                ; ah - reset child's wait channel
4255 0000DB98 89F0               <1>       mov    eax, esi
4256 0000DB9A A2[B3030300]       <1>       mov    [u.uno], al ; child process number
4257                            <1>              ;movb r1,u.uno / set child process number to r1
4258 0000DB9F FE86[AF000300]     <1>         inc     byte [esi+p.stat-1] ; 1, SRUN, 05/02/2014
4259                            <1>             ; incb p.stat-1(r1) / set p.stat entry for child
4260                            <1>                           ; / process to active status
4261                            <1>             ; mov u.ttyp,r2 / put pointer to parent process'
4262                            <1>                           ; / control tty buffer in r2
4263                            <1>                ; beq 2f / branch, if no such tty assigned
4264                            <1>             ; clrb 6(r2) / clear interrupt character in tty buffer
4265                            <1>       ; 2:
4266 0000DBA5 53                 <1>       push   ebx  ; * return address for the child process
4267                            <1>                   ; * Retro UNIX 8086 v1 feature only !
4268                            <1>       ; (Retro UNIX 8086 v1 modification!)
4269                            <1>              ; mov $runq+4,r2
4270 0000DBA6 BB[54030300]       <1>       mov    ebx, runq+2 ; normal run queue ; 02/01/2017
4271 0000DBAB E8983D0000         <1>       call   putlu
4272                            <1>              ; jsr r0,putlu / put child process on lowest priority
4273                            <1>                       ; / run queue
4274 0000DBB0 66D1E6             <1>       shl    si, 1
4275                            <1>              ; asl r1 / multiply r1 by 2 to get index
4276                            <1>                     ; / into p.pid table
4277 0000DBB3 66FF05[4E030300]   <1>       inc    word [mpid]
4278                            <1>              ; inc mpid / increment m.pid; get a new process name
4279 0000DBBA 66A1[4E030300]     <1>       mov    ax, [mpid]
4280 0000DBC0 668986[1E000300]   <1>       mov    [esi+p.pid-2], ax
4281                            <1>              ;mov mpid,p.pid-2(r1) / put new process name
4282                            <1>                                ; / in child process' name slot
4283 0000DBC7 5A                 <1>       pop    edx  ; * return address for the child process
4284                            <1>                   ; * Retro UNIX 8086 v1 feature only !
4285 0000DBC8 5B                 <1>       pop    ebx  ; **
4286                            <1>       ;mov   ebx, [esp] ; ** parent process number
4287                            <1>              ; movb (sp),r2 / put parent process number in r2
4288 0000DBC9 66D1E3             <1>       shl    bx, 1
4289                            <1>              ;asl r2 / multiply by 2 to get index into below tables
4290                            <1>       ;movzx eax, word [ebx+p.pid-2]
4291 0000DBCC 668B83[1E000300]   <1>       mov    ax, [ebx+p.pid-2]
4292                            <1>              ; mov p.pid-2(r2),r2 / get process name of parent
4293                            <1>                                ; / process
4294 0000DBD3 668986[3E000300]   <1>       mov    [esi+p.ppid-2], ax
4295                            <1>              ; mov r2,p.ppid-2(r1) / put parent process name
```

```
4296                                    <1>                     ; / in parent process slot for child
4297 0000DBDA A3[64030300]              <1>       mov   [u.r0], eax
4298                                    <1>             ; mov r2,*u.r0 / put parent process name on stack
4299                                    <1>                       ; / at location where r0 was saved
4300 0000DBDF 8B2D[5C030300]            <1>       mov   ebp, [u.sp] ; points to return address (EIP for IRET)
4301 0000DBE5 895500                    <1>       mov   [ebp], edx ; *, CS:EIP -> EIP
4302                                    <1>                      ; * return address for the child process
4303                                    <1>             ; mov $sysret1,-(sp) /
4304                                    <1>             ; mov sp,u.usp / contents of sp at the time when
4305                                    <1>                         ; / user is swapped out
4306                                    <1>             ; mov $sstack,sp / point sp to swapping stack space
4307                                    <1>       ; 04/09/2015 - 01/09/2015
4308                                    <1>       ; [u.usp] = esp
4309 0000DBE8 68[CCD70000]             <1>       push  sysret ; ***
4310 0000DBED 8925[60030300]            <1>       mov   [u.usp], esp ; points to 'sysret' address (***)
4311                                    <1>                          ; (for child process)
4312 0000DBF3 31C0                      <1>       xor   eax, eax
4313 0000DBF5 66A3[94030300]            <1>       mov   [u.ttyp], ax ; 0
4314                                    <1>       ;
4315 0000DBFB E8D83C0000                <1>       call  wswap ; Retro UNIX 8086 v1 modification !
4316                                    <1>             ;jsr r0,wswap / put child process out on drum
4317                                    <1>             ;jsr r0,unpack / unpack user stack
4318                                    <1>             ;mov u.usp,sp / restore user stack pointer
4319                                    <1>             ; tst (sp)+ / bump stack pointer
4320                                    <1>       ; Retro UNIX 386 v1 modification !
4321 0000DC00 58                        <1>       pop   eax ; ***
4322 0000DC01 66D1E3                    <1>       shl   bx, 1
4323 0000DC04 8B83[BC000300]            <1>       mov     eax, [ebx+p.upage-4] ; UPAGE address ; 14/05/2015
4324 0000DC0A E8013D0000                <1>       call  rswap ; restore parent process 'u' structure,
4325                                    <1>                   ; registers and return address (for IRET)
4326                                    <1>             ;movb (sp)+,u.uno / put parent process number in u.uno
4327 0000DC0F 0FB705[4E030300]          <1>       movzx eax, word [mpid]
4328 0000DC16 A3[64030300]              <1>       mov   [u.r0], eax
4329                                    <1>             ; mov mpid,*u.r0 / put child process name on stack
4330                                    <1>                         ; / where r0 was saved
4331                                    <1>             ; add $2,18.(sp) / add 2 to pc on stack; gives parent
4332                                    <1>                            ; / process return
4333                                    <1>       ;xor   ebx, ebx
4334 0000DC1B 31F6                      <1>       xor     esi, esi
4335                                    <1>             ;clr r1
4336                                    <1> sysfork_4: ; 1: / search u.fp list to find the files
4337                                    <1>                ; / opened by the parent process
4338                                    <1>       ; 01/09/2015
4339                                    <1>       ;xor   bh, bh
4340                                    <1>       ;mov   bl, [esi+u.fp]
4341 0000DC1D 8A86[6A030300]            <1>       mov   al, [esi+u.fp]
4342                                    <1>             ; movb u.fp(r1),r2 / get an open file for this process
4343                                    <1>       ;or    bl, bl
4344 0000DC23 08C0                      <1>       or    al, al
4345 0000DC25 740D                      <1>       jz    short sysfork_5
4346                                    <1>             ; beq 2f / file has not been opened by parent,
4347                                    <1>                    ; / so branch
4348 0000DC27 B40A                      <1>       mov   ah, 10 ; Retro UNIX 386 v1 fsp structure size = 10 bytes
4349 0000DC29 F6E4                      <1>       mul   ah
4350                                    <1>       ;movzx ebx, ax
4351 0000DC2B 6689C3                    <1>       mov   bx, ax
4352                                    <1>       ;shl   bx, 3
4353                                    <1>             ; asl r2 / multiply by 8
4354                                    <1>                   ; asl r2 / to get index into fsp table
4355                                    <1>                   ; asl r2
4356 0000DC2E FE83[4E010300]            <1>       inc   byte [ebx+fsp-2]
4357                                    <1>             ; incb fsp-2(r2) / increment number of processes
4358                                    <1>                         ; / using file, because child will now be
4359                                    <1>                         ; / using this file
4360                                    <1> sysfork_5: ; 2:
4361 0000DC34 46                        <1>       inc   esi
4362                                    <1>             ; inc r1 / get next open file
4363 0000DC35 6683FE0A                  <1>       cmp   si, 10
4364                                    <1>             ; cmp r1,$10. / 10. files is the maximum number which
4365                                    <1>                      ; / can be opened
4366 0000DC39 72E2                      <1>       jb    short sysfork_4
4367                                    <1>             ; blt 1b / check next entry
4368 0000DC3B E98CFBFFFF                <1>       jmp   sysret
4369                                    <1>             ; br sysret1
4370                                    <1>
4371                                    <1> syscreat: ; < create file >
4372                                    <1>       ; 13/11/2017
4373                                    <1>       ; 27/10/2016
4374                                    <1>       ; 25/10/2016, 26/10/2016
4375                                    <1>       ; 15/10/2016, 16/10/2016, 17/10/2016
4376                                    <1>       ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
4377                                    <1>       ;          -derived from INT_21H.ASM-
4378                                    <1>       ;            ("loc_INT21h_create_file")
4379                                    <1>       ;     10/07/2011 (12/03/2011)
4380                                    <1>       ;     INT 21h Function AH = 3Ch
4381                                    <1>       ;     Create File
4382                                    <1>       ;     INPUT
4383                                    <1>       ;        CX = Attributes
4384                                    <1>       ;        DS:DX= Address of zero terminaned path name
4385                                    <1>       ;
4386                                    <1>       ; 27/12/2015 (Retro UNIX 386 v1.1)
4387                                    <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
4388                                    <1>       ; 27/05/2013 (Retro UNIX 8086 v1)
4389                                    <1>       ;
4390                                    <1>       ; 'syscreat' called with two arguments; name and mode.
4391                                    <1>       ; u.namep points to name of the file and mode is put
4392                                    <1>       ; on the stack. 'namei' is called to get i-number of the file.
4393                                    <1>       ; If the file aready exists, it's mode and owner remain
4394                                    <1>       ; unchanged, but it is truncated to zero length. If the file
4395                                    <1>       ; did not exist, an i-node is created with the new mode via
4396                                    <1>       ; 'maknod' whether or not the file already existed, it is
4397                                    <1>       ; open for writing. The fsp table is then searched for a free
4398                                    <1>       ; entry. When a free entry is found, proper data is placed
4399                                    <1>       ; in it and the number of this entry is put in the u.fp list.
4400                                    <1>       ; The index to the u.fp (also know as the file descriptor)
```

```
4401                                    <1>       ; is put in the user's r0.
4402                                    <1>       ;
4403                                    <1>       ; Calling sequence:
4404                                    <1>       ;     syscreate; name; mode
4405                                    <1>       ; Arguments:
4406                                    <1>       ;     name - name of the file to be created
4407                                    <1>       ;     mode - mode of the file to be created
4408                                    <1>       ; Inputs: (arguments)
4409                                    <1>       ; Outputs: *u.r0 - index to u.fp list
4410                                    <1>       ;              (the file descriptor of new file)
4411                                    <1>       ; ..........................................................
4412                                    <1>       ;
4413                                    <1>       ; Retro UNIX 8086 v1 modification:
4414                                    <1>       ;      'syscreate' system call has two arguments; so,
4415                                    <1>       ;     * 1st argument, name is pointed to by BX register
4416                                    <1>       ;     * 2nd argument, mode is in CX register
4417                                    <1>       ;
4418                                    <1>       ;     AX register (will be restored via 'u.r0') will return
4419                                    <1>       ;     to the user with the file descriptor/number
4420                                    <1>       ;     (index to u.fp list).
4421                                    <1>       ;
4422                                    <1>       ;call  arg2
4423                                    <1>       ; * name - 'u.namep' points to address of file/path name
4424                                    <1>       ;          in the user's program segment ('u.segmnt')
4425                                    <1>       ;          with offset in BX register (as sysopen argument 1).
4426                                    <1>       ; * mode - sysopen argument 2 is in CX register
4427                                    <1>       ;          which is on top of stack.
4428                                    <1>       ;
4429                                    <1>       ; TRDOS 386 (10/10/2016)
4430                                    <1>       ;
4431                                    <1>       ; INPUT ->
4432                                    <1>       ;        CL = File Attributes
4433                                    <1>       ;             bit 0 (1) - Read only file (R)
4434                                    <1>       ;             bit 1 (1) - Hidden file (H)
4435                                    <1>       ;             bit 2 (1) - System file (R)
4436                                    <1>       ;             bit 3 (1) - Volume label/name (V)
4437                                    <1>       ;             bit 4 (1) - Subdirectory (D)
4438                                    <1>       ;           bit 5 (1) - File has been archived (A)
4439                                    <1>       ;        EBX = Pointer to filename (ASCIIZ) -path-
4440                                    <1>       ;
4441                                    <1>       ; OUTPUT ->
4442                                    <1>       ;          eax = File/Device Handle/Number (index) (AL)
4443                                    <1>       ;          cf = 1 -> Error code in AL
4444                                    <1>       ;
4445                                    <1>       ; Modified Registers: EAX (at the return of system call)
4446                                    <1>       ;
4447                                    <1>       ; Note: If the file is existing and it has not any one
4448                                    <1>       ;     of S,H,R,V,D attributes, it will be truncated
4449                                    <1>       ;     to zero length; otherwise, access error will be
4450                                    <1>       ;     returned.
4451                                    <1>
4452                                    <1> sysmkdir_0:
4453 0000DC40 F6C108                   <1>       test   cl, 08h ; Volume name
4454 0000DC43 740A                     <1>       jz     short syscreat_0
4455                                    <1>
4456                                    <1>       ; Volume name or long name creation
4457                                    <1>       ; is not permitted (in TRDOS 386)!
4458 0000DC45 B80B000000               <1>       mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
4459 0000DC4A E926020000               <1>         jmp  sysopen_dev_err
4460                                    <1>
4461                                    <1> syscreat_0:
4462                                    <1>       ;mov   [u.namep], ebx
4463 0000DC4F 51                       <1>       push   ecx
4464 0000DC50 89DE                     <1>       mov    esi, ebx
4465                                    <1>       ; file name is forced, change directory as temporary
4466                                    <1>       ;mov   ax, 1
4467                                    <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
4468                                    <1>       ;call  set_working_path
4469 0000DC52 E849520000               <1>       call   set_working_path_x ; 17/10/2016
4470 0000DC57 0F82D7000000             <1>       jc     syscreat_err
4471                                    <1>
4472                                    <1>       ; 16/10/2016
4473 0000DC5D 803D[A7950100]00         <1>       cmp    byte [SWP_inv_fname], 0
4474 0000DC64 776C                     <1>       ja     short  syscreat_inv_fname ; invalid file name !
4475                                    <1>
4476                                    <1>       ; Here, we have a valid path and also a valid file name
4477                                    <1>       ; (Working dir has been changed if the path
4478                                    <1>       ;  -file name string- had contained a dir name.)
4479                                    <1>
4480 0000DC66 6631C0                   <1>       xor    ax, ax
4481                                    <1>       ;mov   esi, FindFile_Name
4482 0000DC69 E8E3B6FFFF               <1>       call   find_first_file
4483 0000DC6E 59                       <1>       pop    ecx
4484                                    <1>              ; ESI = Directory Entry (FindFile_DirEntry) Location
4485                                    <1>              ; EDI = Directory Buffer Directory Entry Location
4486                                    <1>              ; EAX = File Size
4487                                    <1>              ;  BL = Attributes of The File/Directory
4488                                    <1>              ;  BH = Long Name Yes/No Status (>0 is YES)
4489                                    <1>              ;  DX > 0 : Ambiguous filename chars are used
4490 0000DC6F 7269                     <1>       jc     short syscreat_1 ; file not found (the good!)
4491                                    <1>                             ; or another error (the bad')
4492                                    <1>
4493                                    <1>       ; (& the uggly!) truncate file to zero length before open
4494                                    <1>
4495                                    <1>       ;'*' and '?' already checked at 'set_working_path' stage
4496                                    <1>       ;and   dx, dx
4497                                    <1>       ;jnz   short sysmkdir_err ; permission denied
4498                                    <1>                               ; invalid filename chars
4499                                    <1>
4500                                    <1>       ;test  cl, 10h ; subdirectory ?
4501                                    <1>       ;jnz   short sysmkdir_err
4502                                    <1>
4503                                    <1>       ; BL = File Attributes:
4504                                    <1>       ;           bit 0 (1) - Read only file (R)
4505                                    <1>       ;           bit 1 (1) - Hidden file (H)
```

```
4506                                    <1>       ;            bit 2 (1) - System file (R)
4507                                    <1>       ;            bit 3 (1) - Volume label/name (V)
4508                                    <1>       ;            bit 4 (1) - Subdirectory (D)
4509                                    <1>       ;           bit 5 (1) - File has been archived
4510                                    <1>
4511                                    <1>       ; * existing directory must not be truncated
4512                                    <1>       ;   (we don't know it is empty or not, at this stage)
4513                                    <1>       ; * existing volume name (or a long name) can not be
4514                                    <1>       ;   re-created or truncated by 'syscreat'
4515                                    <1>       ; * A file with S, H, R attributes must not be truncated
4516                                    <1>       ;   (change attributes to normal, if you need truncate it)
4517                                    <1>
4518 0000DC71 F6C31F                   <1>       test   bl, 00011111b  ; check attributes of existing file
4519 0000DC74 754E                     <1>       jnz    short sysmkdir_err
4520                                    <1>
4521                                    <1>       ;; normal file, OK to continue...
4522                                    <1>
4523                                    <1>       ; ESI = FindFile_DirEntry
4524 0000DC76 668B4614                 <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
4525 0000DC7A C1E010                   <1>       shl    eax, 16 ; 13/11/2017
4526 0000DC7D 668B461A                 <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
4527                                    <1>       ; EAX = First cluster to be truncated/unlinked
4528 0000DC81 57                       <1>       push   edi
4529 0000DC82 51                       <1>       push   ecx
4530 0000DC83 BE00010900               <1>       mov    esi, Logical_DOSDisks
4531 0000DC88 29C9                     <1>       sub    ecx, ecx
4532 0000DC8A 8A2D[B6890100]           <1>       mov    ch, [Current_Drv]
4533 0000DC90 01CE                     <1>       add    esi, ecx
4534                                    <1>       ; ESI = Logical dos drive description table address
4535 0000DC92 E8C9F7FFFF               <1>       call   truncate_cluster_chain
4536 0000DC97 59                       <1>       pop    ecx
4537 0000DC98 5F                       <1>       pop    edi
4538 0000DC99 7230                     <1>       jc     short syscreate_truncate_err
4539                                    <1>
4540                                    <1>       ; 26/10/2016
4541                                    <1>       ; EDI = Directory entry address in directory buffer
4542                                    <1>       ; Update directory entry
4543 0000DC9B E848DCFFFF               <1>       call   convert_current_date_time
4544                                    <1>       ; OUTPUT -> DX = Date in dos dir entry format
4545                                    <1>       ;         AX = Time in dos dir entry format
4546 0000DCA0 66894716                 <1>       mov    [edi+DirEntry_WrtTime], ax
4547 0000DCA4 66895718                 <1>       mov    [edi+DirEntry_WrtDate], dx
4548 0000DCA8 66895712                 <1>       mov    [edi+DirEntry_LastAccDate], dx
4549 0000DCAC 31C0                     <1>       xor    eax, eax ; file size = 0
4550 0000DCAE 89471C                   <1>       mov    [edi+DirEntry_FileSize], eax ; 0
4551 0000DCB1 C605[DC900100]02         <1>       mov    byte [DirBuff_ValidData], 2 ; data changed sign
4552 0000DCB8 BE[A8920100]             <1>       mov    esi, FindFile_DirEntry
4553 0000DCBD B201                     <1>       mov    dl, 1 ; open file for writing
4554 0000DCBF E9AA000000               <1>       jmp    sysopen_2
4555                                    <1>
4556                                    <1> sysmkdir_err:
4557                                    <1>       ; 1 = write, 2 = read & write, >2 = invalid
4558 0000DCC4 B80B000000               <1>       mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
4559 0000DCC9 EB73                     <1>       jmp    short sysopen_err
4560                                    <1>
4561                                    <1> syscreate_truncate_err:
4562 0000DCCB B812000000               <1>       mov    eax, ERR_DRV_WRITE ; 18 ; 'disk write error !'
4563 0000DCD0 EB6C                     <1>       jmp    short sysopen_err
4564                                    <1>
4565                                    <1> syscreat_inv_fname:  ; invalid file name chars
4566                                    <1>       ; 16/10/2016
4567 0000DCD2 B81A000000               <1>       mov    eax, ERR_INV_FILE_NAME  ; 26 ; invalid file name chars
4568 0000DCD7 59                       <1>       pop    ecx
4569 0000DCD8 EB64                     <1>       jmp    sysopen_err
4570                                    <1>
4571                                    <1> syscreat_1:
4572                                    <1>       ; Error code in EAX
4573 0000DCDA 3C02                     <1>       cmp    al, 02h ; 'File not found' error
4574 0000DCDC 7560                     <1>       jne    sysopen_err
4575                                    <1>
4576 0000DCDE F6C110                   <1>       test   cl, 10h ; Directory
4577 0000DCE1 0F852C020000             <1>       jnz    sysmkdir_2
4578                                    <1>
4579                                    <1> syscreat_2:
4580 0000DCE7 BE[98920100]             <1>       mov    esi, FindFile_Name
4581                                    <1>       ;xor   edx, edx
4582 0000DCEC 31C0                     <1>       xor    eax, eax ; File Size  = 0
4583 0000DCEE 31DB                     <1>       xor    ebx, ebx
4584 0000DCF0 4B                       <1>       dec    ebx ; FFFFFFFFh -> create empty file
4585                                    <1>                  ;             (only for FAT fs)
4586                                    <1>       ; CL = File Attributes
4587 0000DCF1 E8F8EBFFFF               <1>       call   create_file
4588 0000DCF6 7246                     <1>       jc     sysopen_err
4589                                    <1>              ; EAX = New file's first cluster
4590                                    <1>              ; ESI = Logical Dos Drv Descr. Table Addr.
4591                                    <1>              ; EBX = offset CreateFile_Size
4592                                    <1>              ; ECX = Sectors per cluster (<256)
4593                                    <1>              ; EDX = Directory entry index/number (<65536)
4594                                    <1>       ; 26/10/2016
4595                                    <1>       ;mov   esi, Directory_Buffer
4596                                    <1>       ;shl   dx, 5 ; *32
4597                                    <1>       ;add   esi, edx
4598                                    <1>       ;; esi = directory entry address in directory buffer
4599                                    <1>
4600                                    <1>       ; Here, directory entry has been created but last
4601                                    <1>       ; modification date & time of the parent dir has not
4602                                    <1>       ; been updated, yet!
4603                                    <1>       ; (Note: Directory and FAT buffers have been updated...)
4604                                    <1>
4605 0000DCF8 E824DDFFFF               <1>       call   update_parent_dir_lmdt ; now, it is OK too!
4606                                    <1>
4607                                    <1>       ; 25/10/2016
4608 0000DCFD 66B80018                 <1>       mov    ax, 1800h
4609 0000DD01 BE[98920100]             <1>       mov    esi, FindFile_Name
4610 0000DD06 E846B6FFFF               <1>       call   find_first_file
```

```
4611 0000DD0B 7231              <1>       jc    short sysopen_err
4612                            <1>
4613                            <1>       ; Only possible error after here is
4614                            <1>       ; "too many open files !" error.
4615                            <1>       ;
4616                            <1>       ; If "syscreat" will return with that error,
4617                            <1>       ; (the file has been created but it could not be opened)
4618                            <1>       ; the user must retry to open this file again
4619                            <1>       ; or must close another file before using
4620                            <1>       ; "sysopen" system call.
4621                            <1>
4622 0000DD0D B201              <1>       mov   dl, 1 ; open file for writing
4623                            <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
4624                            <1>       ; EAX = File Size (= 0)
4625 0000DD0F EB5D              <1>       jmp   short sysopen_2
4626                            <1>
4627                            <1> sysopen: ;<open file>
4628                            <1>       ; 26/10/2016
4629                            <1>       ; 24/10/2016
4630                            <1>       ; 17/10/2016
4631                            <1>       ; 15/10/2016
4632                            <1>       ; 06/10/2016, 07/10/2016, 08/10/2016
4633                            <1>       ; 05/10/2016 (TRDOS 386 = TRDOS v2.0)
4634                            <1>       ;          -derived from INT_21H.ASM-
4635                            <1>       ;            ("loc_INT21h_open_file")
4636                            <1>       ;     26/02/2011
4637                            <1>       ;     INT 21h Function AH = 3Dh
4638                            <1>       ;     Open File
4639                            <1>       ;     INPUT
4640                            <1>       ;        AL= File Access Value
4641                            <1>       ;          0- Open for reading
4642                            <1>       ;            1- Open for writing
4643                            <1>       ;            2- Open for reading and writing
4644                            <1>       ;        DS:DX= Pointer to filename (ASCIIZ)
4645                            <1>       ;
4646                            <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
4647                            <1>       ; 22/05/2013 - 27/05/2013 (Retro UNIX 8086 v1)
4648                            <1>       ;
4649                            <1>       ; 'sysopen' opens a file in following manner:
4650                            <1>       ;    1) The second argument in a sysopen says whether to
4651                            <1>       ;     open the file ro read (0) or write (>0).
4652                            <1>       ;    2) I-node of the particular file is obtained via 'namei'.
4653                            <1>       ;    3) The file is opened by 'iopen'.
4654                            <1>       ;    4) Next housekeeping is performed on the fsp table
4655                            <1>       ;      and the user's open file list - u.fp.
4656                            <1>       ;      a) u.fp and fsp are scanned for the next available slot.
4657                            <1>       ;      b) An entry for the file is created in the fsp table.
4658                            <1>       ;      c) The number of this entry is put on u.fp list.
4659                            <1>       ;      d) The file descriptor index to u.fp list is pointed
4660                            <1>       ;         to by u.r0.
4661                            <1>       ;
4662                            <1>       ; Calling sequence:
4663                            <1>       ;     sysopen; name; mode
4664                            <1>       ; Arguments:
4665                            <1>       ;     name - file name or path name
4666                            <1>       ;     mode - 0 to open for reading
4667                            <1>       ;            1 to open for writing
4668                            <1>       ; Inputs: (arguments)
4669                            <1>       ; Outputs: *u.r0 - index to u.fp list (the file descriptor)
4670                            <1>       ;              is put into r0's location on the stack.
4671                            <1>       ; ..........................................................
4672                            <1>       ;
4673                            <1>       ; Retro UNIX 8086 v1 modification:
4674                            <1>       ;       'sysopen' system call has two arguments; so,
4675                            <1>       ;     * 1st argument, name is pointed to by BX register
4676                            <1>       ;     * 2nd argument, mode is in CX register
4677                            <1>       ;
4678                            <1>       ;     AX register (will be restored via 'u.r0') will return
4679                            <1>       ;     to the user with the file descriptor/number
4680                            <1>       ;     (index to u.fp list).
4681                            <1>       ;
4682                            <1>       ;call  arg2
4683                            <1>       ; * name - 'u.namep' points to address of file/path name
4684                            <1>       ;          in the user's program segment ('u.segmnt')
4685                            <1>       ;          with offset in BX register (as sysopen argument 1).
4686                            <1>       ; * mode - sysopen argument 2 is in CX register
4687                            <1>       ;          which is on top of stack.
4688                            <1>       ;
4689                            <1>       ; jsr r0,arg2 / get sys args into u.namep and on stack
4690                            <1>       ;
4691                            <1>            ; system call registers: ebx, ecx (through 'sysenter')
4692                            <1>       ;
4693                            <1>       ; TRDOS 386 (05/10/2016)
4694                            <1>       ;
4695                            <1>       ; INPUT ->
4696                            <1>       ;        CL = File Access Value (Open Mode)
4697                            <1>       ;           0 - Open file for reading
4698                            <1>       ;             1 - Open file for writing
4699                            <1>       ;             2 - Open device for reading
4700                            <1>       ;           3 - Open device for writing
4701                            <1>       ;          EBX = Pointer to filename/devicename (ASCIIZ)
4702                            <1>       ; OUTPUT ->
4703                            <1>       ;          eax = File/Device Handle/Number (index) (AL)
4704                            <1>       ;          cf = 1 -> Error code in AL
4705                            <1>       ;
4706                            <1>       ; Modified Registers: EAX (at the return of system call)
4707                            <1>       ;
4708                            <1>
4709 0000DD11 80F901            <1>       cmp   cl, 1 ; read file (0), write file (1)
4710 0000DD14 7614              <1>       jna   short sysopen_0
4711                            <1>
4712 0000DD16 80F903            <1>       cmp   cl, 3
4713 0000DD19 0F8640010000      <1>       jna   sysopen_device
4714                            <1>
4715                            <1>       ; Invalid access code
```

```
4716 0000DD1F B817000000          <1>       mov   eax, ERR_INV_PARAMETER
4717 0000DD24 0F874B010000        <1>       ja    sysopen_dev_err
4718                              <1>
4719                              <1> sysopen_0:
4720                              <1>       ;mov   [u.namep], ebx
4721 0000DD2A 51                  <1>       push  ecx
4722 0000DD2B 89DE                <1>       mov   esi, ebx
4723                              <1>       ; file name is forced, change directory as temporary
4724                              <1>       ;mov   ax, 1
4725                              <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
4726                              <1>       ;call  set_working_path
4727 0000DD2D E86E510000          <1>       call  set_working_path_x ; 17/10/2016
4728 0000DD32 731E                <1>       jnc   short sysopen_1
4729                              <1>
4730                              <1> syscreat_err: ; ecx = file attributes (for 'syscreat')
4731 0000DD34 59                  <1>       pop   ecx ; open mode
4732 0000DD35 21C0                <1>       and   eax, eax  ; 0 -> Bad Path!
4733 0000DD37 7505                <1>       jnz   short sysopen_err
4734                              <1>       ; eax = 0
4735 0000DD39 B80C000000          <1>       mov   eax, ERR_DIR_NOT_FOUND ; Directory not found !
4736                              <1> sysopen_err:
4737 0000DD3E A3[64030300]        <1>       mov   [u.r0], eax
4738 0000DD43 A3[C8030300]        <1>       mov   [u.error], eax
4739 0000DD48 E828520000          <1>       call  reset_working_path
4740 0000DD4D E95AFAFFFF          <1>       jmp   error
4741                              <1>
4742                              <1> sysopen_1:
4743                              <1>       ;mov   esi, FindFile_Name
4744 0000DD52 66B80018            <1>       mov   ax, 1800h ; Only files
4745 0000DD56 E8F6B5FFFF          <1>       call  find_first_file
4746 0000DD5B 5A                  <1>       pop   edx
4747 0000DD5C 72E0                <1>       jc    short sysopen_err ; eax = 2 (File not found !)
4748                              <1>
4749                              <1>       ; check_open_file_attr_access_code
4750                              <1>
4751 0000DD5E F6C307              <1>       test  bl, 7  ; system, hidden, readonly
4752 0000DD61 740B                <1>       jz    short sysopen_2
4753                              <1>
4754 0000DD63 20D2                <1>       and   dl, dl ; 0 = read mode
4755 0000DD65 7407                <1>       jz    short sysopen_2
4756                              <1>
4757                              <1>       ; 1 = write, 2 = read & write, >2 = invalid
4758 0000DD67 B80B000000          <1>       mov   eax, ERR_FILE_ACCESS  ; 11 = 'permission denied !'
4759 0000DD6C EBD0                <1>       jmp   short sysopen_err
4760                              <1>
4761                              <1> sysopen_2:
4762                              <1>       ; esi = Directory Entry (FindFile_DirEntry) Location
4763 0000DD6E 89F3                <1>       mov   ebx, esi
4764 0000DD70 31F6                <1>       xor   esi, esi ; 0
4765 0000DD72 31FF                <1>       xor   edi, edi ; 0
4766                              <1> sysopen_3: ; scan the list of entries in fsp table
4767 0000DD74 80BE[6A030300]00    <1>       cmp   byte [esi+u.fp], 0
4768 0000DD7B 760F                <1>       jna   short sysopen_4 ; empty slot
4769 0000DD7D 6646                <1>       inc   si
4770 0000DD7F 6683FE0A            <1>       cmp   si, 10
4771 0000DD83 72EF                <1>       jb    short sysopen_3
4772                              <1> toomanyf:
4773 0000DD85 B80D000000          <1>       mov   eax, ERR_TOO_MANY_FILES ; too many open files !
4774 0000DD8A EBB2                <1>       jmp   short sysopen_err
4775                              <1>
4776                              <1> sysopen_4:
4777 0000DD8C 80BF[16990100]00    <1>       cmp    byte [edi+OF_MODE], 0 ; Scan open files table
4778 0000DD93 760A                <1>       jna    short sysopen_5
4779 0000DD95 6647                <1>       inc    di
4780 0000DD97 6683FF0A            <1>       cmp    di, OPENFILES ; max. number of open files (=10)
4781 0000DD9B 72EF                <1>       jb     short sysopen_4
4782 0000DD9D EBE6                <1>       jmp    short toomanyf
4783                              <1>
4784                              <1> sysopen_5:
4785 0000DD9F FEC2                <1>       inc    dl
4786 0000DDA1 8897[16990100]      <1>       mov   [edi+OF_MODE], dl
4787 0000DDA7 8A15[56920100]      <1>       mov    dl, [FindFile_Drv]
4788 0000DDAD 8897[0C990100]      <1>       mov   [edi+OF_DRIVE], dl ; Logical DOS drive number
4789 0000DDB3 66C1E702            <1>       shl    di, 2 ; *4 (dword offset)
4790                              <1>
4791 0000DDB7 8987[5C990100]      <1>       mov    [edi+OF_SIZE], eax ; File size in bytes
4792                              <1>
4793 0000DDBD 668B4314            <1>       mov    ax, [ebx+DirEntry_FstClusHI]
4794 0000DDC1 C1E010              <1>       shl    eax, 16
4795 0000DDC4 668B431A            <1>       mov    ax, [ebx+DirEntry_FstClusLO]
4796 0000DDC8 8987[E4980100]      <1>       mov   [edi+OF_FCLUSTER], eax ; First cluster
4797 0000DDCE 8987[FC990100]      <1>       mov   [edi+OF_CCLUSTER], eax ; Current cluster
4798                              <1>
4799 0000DDD4 31DB                <1>       xor    ebx, ebx
4800 0000DDD6 899F[34990100]      <1>       mov   [edi+OF_POINTER], ebx ; offset pointer (0)
4801 0000DDDC 899F[249A0100]      <1>       mov   [edi+OF_CCINDEX], ebx ; cluster index (0)
4802                              <1>
4803 0000DDE2 A1[C8920100]        <1>       mov    eax, [FindFile_DirFirstCluster]
4804 0000DDE7 8987[84990100]      <1>       mov    [edi+OF_DIRFCLUSTER], eax
4805                              <1>
4806 0000DDED A1[CC920100]        <1>       mov    eax, [FindFile_DirCluster]
4807 0000DDF2 8987[AC990100]      <1>       mov    [edi+OF_DIRCLUSTER], eax
4808                              <1>
4809                              <1>       ; Get (& Save) Volume ID
4810                              <1>       ; Important for files of removable drives
4811                              <1>       ; (In order to check the drive has same volume/disk)
4812 0000DDF8 88D7                <1>       mov    bh, dl
4813 0000DDFA 81C300010900        <1>       add    ebx, Logical_DOSDisks
4814 0000DE00 8A4303              <1>       mov    al, [ebx+LD_FATType]
4815 0000DE03 3C01                <1>       cmp    al, 1
4816 0000DE05 7209                <1>       jb     short sysopen_6_fs
4817 0000DE07 3C02                <1>       cmp    al, 2
4818 0000DE09 770A                <1>       ja     short sysopen_6_fat32
4819                              <1> sysopen_6_fat:
4820 0000DE0B 8B432D              <1>       mov    eax, [ebx+LD_BPB+VolumeID]
```

```
4821 0000DE0E EB08              <1>       jmp    short sysopen_7
4822                            <1> sysopen_6_fs:
4823 0000DE10 8B4328            <1>       mov    eax, [ebx+LD_FS_VolumeSerial]
4824 0000DE13 EB03              <1>       jmp    short sysopen_7
4825                            <1> sysopen_6_fat32:
4826 0000DE15 8B4349            <1>       mov    eax, [ebx+LD_BPB+FAT32_VolID]
4827                            <1> sysopen_7:
4828 0000DE18 A3[AC890100]      <1>       mov    [Current_VolSerial], eax
4829                            <1>
4830 0000DE1D 8987[D4990100]    <1>       mov    [edi+OF_VOLUMEID], eax
4831                            <1>
4832                            <1>       ; 24/10/2016
4833 0000DE23 66D1EF            <1>       shr    di, 1 ; 4/2, word offset
4834 0000DE26 668B1D[D0920100]  <1>       mov    bx, [FindFile_DirEntryNumber]
4835 0000DE2D 66899F[4C9A0100]  <1>       mov    [edi+OF_DIRENTRY], bx
4836                            <1>
4837 0000DE34 31D2              <1>       xor    edx, edx
4838                            <1>       ;shr   di, 2 ; /4 (byte offset)
4839 0000DE36 66D1EF            <1>       shr    di, 1 ; 2/2, byte offset
4840 0000DE39 8897[2A990100]    <1>       mov    byte [edi+OF_OPENCOUNT], dl ; 0
4841 0000DE3F 8897[20990100]    <1>       mov    byte [edi+OF_STATUS], dl ; 0
4842                            <1>
4843 0000DE45 89FB              <1>       mov    ebx, edi
4844 0000DE47 FEC3              <1>       inc    bl
4845                            <1>
4846 0000DE49 889E[6A030300]    <1>       mov    [esi+u.fp], bl ; Open File Entry Number
4847 0000DE4F 8935[64030300]    <1>       mov    [u.r0], esi ; move index to u.fp list
4848                            <1>                          ; into eax on stack
4849                            <1>
4850 0000DE55 E81B510000        <1>       call   reset_working_path
4851                            <1>
4852 0000DE5A E96DF9FFFF        <1>       jmp    sysret
4853                            <1>
4854                            <1>       ; (Retro UNIX 386 v1.0)
4855                            <1>       ; 'fsp' table (10 bytes/entry)
4856                            <1>       ; bit 15                              bit 0
4857                            <1>       ; ---|----------------------------------------
4858                            <1>       ; r/w|       i-number of open file
4859                            <1>       ; ---|----------------------------------------
4860                            <1>       ;               device number
4861                            <1>       ; ----------------------------------------------
4862                            <1>       ; offset pointer, r/w pointer to file (bit 0-15)
4863                            <1>       ; ----------------------------------------------
4864                            <1>       ; offset pointer, r/w pointer to file (bit 16-31)
4865                            <1>       ; ---------------------|------------------------
4866                            <1>       ;  flag that says file    | number of processes
4867                            <1>       ;   has been deleted      | that have file open
4868                            <1>       ; ---------------------|------------------------
4869                            <1>
4870                            <1> sysopen_device:
4871                            <1>       ; 15/10/2016
4872                            <1>       ; 08/10/2016
4873                            <1>       ; 07/10/2016 (TRDOS 386 = TRDOS v2.0)
4874 0000DE5F 51                <1>       push   ecx ; open mode
4875 0000DE60 89E5              <1>       mov    ebp, esp
4876 0000DE62 B910000000        <1>       mov    ecx, 16 ; transfer length = 16 bytes
4877 0000DE67 29CC              <1>       sub    esp, ecx
4878 0000DE69 89E7              <1>       mov    edi, esp ; destination address
4879 0000DE6B 89DE              <1>       mov    esi, ebx ; dev name in user's memory space
4880 0000DE6D E8FC3B0000        <1>       call   transfer_from_user_buffer
4881 0000DE72 7310              <1>       jnc    short sysopen_dev_0
4882                            <1>       ; eax = ERR_OUT_OF_MEMORY = 4 = ERR_MINOR_IM
4883 0000DE74 59                <1>       pop    ecx
4884                            <1> sysopen_dev_err:
4885 0000DE75 A3[64030300]      <1>       mov    [u.r0], eax
4886 0000DE7A A3[C8030300]      <1>       mov    [u.error], eax
4887 0000DE7F E928F9FFFF        <1>       jmp    error
4888                            <1> sysopen_dev_0:
4889 0000DE84 89FE              <1>       mov    esi, edi ; Device name addr (max. 16 bytes, ASCIIZ)
4890                            <1>                       ; for example: "tty, TTY, /dev/tty"
4891 0000DE86 E890530000        <1>       call   get_device_number
4892 0000DE8B 89EC              <1>       mov    esp, ebp
4893 0000DE8D 59                <1>       pop    ecx
4894 0000DE8E 7307              <1>       jnc    short sysopen_dev_1
4895 0000DE90 B818000000        <1>       mov    eax, ERR_INV_DEV_NAME ; 24 ; 'invalid device name !'
4896 0000DE95 EBDE              <1>       jmp    short sysopen_dev_err
4897                            <1> sysopen_dev_1:
4898                            <1>       ; eax = Device Number (AL)
4899                            <1>       ;  cl = Open mode (2 = device read, 3 = device write)
4900 0000DE97 31DB              <1>         xor     ebx, ebx ; 0
4901                            <1> sysopen_dev_2: ; scan the list of entries
4902 0000DE99 389B[6A030300]    <1>       cmp    [ebx+u.fp], bl ; 0
4903 0000DE9F 760E              <1>       jna    short sysopen_dev_3 ; empty slot
4904 0000DEA1 FEC3              <1>       inc    bl
4905 0000DEA3 80FB0A            <1>       cmp    bl, 10
4906 0000DEA6 72F1              <1>       jb     short sysopen_dev_2
4907                            <1>       ;
4908 0000DEA8 B80D000000        <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
4909 0000DEAD EBC6              <1>       jmp    short sysopen_dev_err
4910                            <1> sysopen_dev_3:
4911 0000DEAF 891D[64030300]    <1>       mov    [u.r0], ebx ; File/Device index/handle/descriptor
4912                            <1>       ; eax = device number (entry offset)
4913 0000DEB5 8AA8[A8960100]    <1>       mov    ch, [eax+DEV_ACCESS] ; bit 0 = accessable by users
4914                            <1>                                   ; bit 1 = read access perm
4915                            <1>                                   ; bit 2 = write access perm
4916                            <1>                                   ; bit 3 = IOCTL permit to users
4917                            <1>                                   ; bit 4 = block device if set
4918                            <1>                                   ; bit 5 = 16 bit or 1024 byte
4919                            <1>                                   ; bit 6 = 32 bit or 2048 byte
4920                            <1>                                   ; bit 7 = installable device drv
4921 0000DEBB F6C501              <1>       test   ch, 1 ; accessable by normal users (except root)
4922 0000DEBE 7510              <1>       jnz    short sysopen_dev_4 ; yes, permission has been given
4923 0000DEC0 803D[B0030300]00  <1>       cmp    byte [u.uid], 0 ; root?
4924 0000DEC7 7607              <1>       jna    short sysopen_dev_4 ; superuser can open all devices
4925                            <1> sysopen_dev_perm_err:
```

```
0000DEC9 B80B000000          <1>       mov    eax, ERR_DEV_ACCESS  ; 11 = 'permission denied !'
0000DECE EBA5                <1>       jmp    short sysopen_dev_err
                             <1> sysopen_dev_4:
0000DED0 D0ED                <1>       shr    ch, 1 ; result: 1 = read, 2 = write, 3 = r & w
0000DED2 FEC9                <1>       dec    cl  ; result: 1 = read, 2 = write
0000DED4 84E9                <1>       test   cl, ch
0000DED6 74F1                <1>       jz     short sysopen_dev_perm_err
                             <1>
0000DED8 D0E5                <1>       shl    ch, 1 ; bit 0 = 0
                             <1>       ; eax = device number (entry offset)
0000DEDA E858540000          <1>       call   device_open
0000DEDF 72E8                <1>       jc     short sysopen_dev_perm_err
                             <1>
                             <1>       ; eax = device number (entry offset)
0000DEE1 0C80                <1>       or     al, 80h ; set device bit (set bit 7 to 1)
0000DEE3 8B1D[64030300]      <1>       mov    ebx, [u.r0]
0000DEE9 8883[6A030300]      <1>       mov    [ebx+u.fp], al      ; bit 7 (=1) points to device
                             <1>
0000DEEF E9D8F8FFFF          <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>       ; 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>
0000DEF4 6621C9              <1>       and    cx, cx ; if cx = 0 -> create a normal subdir
0000DEF7 7413                <1>       jz     short sysmkdir_1
                             <1>
0000DEF9 F6C110              <1>       test   cl, 10h ; if dir flags set, also use other flags
0000DEFC 0F853EFDFFFF        <1>       jnz    sysmkdir_0 ; jump to head of 'syscreat'
                             <1>
                             <1>       ; CX has wrong flags
0000DF02 B817000000          <1>       mov    eax, ERR_INV_FLAGS
0000DF07 E969FFFFFF          <1>       jmp    sysopen_dev_err
                             <1>
                             <1> sysmkdir_1:
0000DF0C B110                <1>       mov    cl, 10h ; set subdir flag and reset other flags
0000DF0E E92DFDFFFF          <1>       jmp    sysmkdir_0 ; jump to head of 'syscreat'
                             <1> sysmkdir_2:
                             <1>       ; jump from 'syscreat' ; from 'syscreat_1'
                             <1>       ;  CL = Directory attributes/flags
0000DF13 BE[98920100]        <1>       mov    esi, FindFile_Name
0000DF18 E804D7FFFF          <1>       call   make_sub_directory
0000DF1D 0F821BFEFFFF        <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>                             ; 'sysdefss' ; 10/10/2016
                                <1>
0000DF23 A3[64030300]           <1>       mov   [u.r0], eax ; New sub dir's first cluster
                                <1>
0000DF28 E848500000             <1>       call  reset_working_path
                                <1>
0000DF2D E99AF8FFFF             <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>
0000DF32 89D8                   <1>       mov   eax, ebx
0000DF34 31DB                   <1>       xor   ebx, ebx
0000DF36 891D[64030300]         <1>       mov   [u.r0], ebx ; 0  ; return value of EAX
0000DF3C E8BC2F0000             <1>       call  fclose
0000DF41 0F8385F8FFFF           <1>       jnc   sysret
0000DF47 B80A000000             <1>       mov   eax, ERR_FILE_NOT_OPEN ; file not open !
0000DF4C A3[C8030300]           <1>       mov   [u.error], eax ;
0000DF51 A3[64030300]           <1>       mov   [u.r0], eax ; ! invalid handle !
0000DF56 E951F8FFFF             <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
0000DF5B E8EB2F0000             <1>       call  getf1
0000DF60 7277                   <1>       jc    short device_read ; read data from device
                                <1>       ; EAX = First cluster of the file
                                <1>
0000DF62 E83F000000             <1>       call  rw1
0000DF67 730A                   <1>       jnc   short sysread_0
                                <1>
0000DF69 A3[64030300]           <1>       mov   [u.r0], eax ; error code
0000DF6E E939F8FFFF             <1>       jmp   error
                                <1>
                                <1> sysread_0:
0000DF73 E8E2350000             <1>       call  readi
0000DF78 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
0000DF7A E8CC2F0000             <1>       call  getf1
0000DF7F 7274                   <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>
0000DF81 E820000000             <1>       call  rw1
0000DF86 730A                   <1>       jnc   short syswrite_0
0000DF88 A3[64030300]           <1>       mov   [u.r0], eax ; error code
0000DF8D E91AF8FFFF             <1>       jmp   error
                                <1>
                                <1> syswrite_0:
0000DF92 E8EF3C0000             <1>       call  writei
                                <1> rw0: ; 1:
0000DF97 A1[8C030300]           <1>       mov   eax, [u.nread]
0000DF9C A3[64030300]           <1>       mov   [u.r0], eax
0000DFA1 E926F8FFFF             <1>       jmp   sysret
                                <1>
                                <1> rw1:
                                <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 14/05/2015 (Retro UNIX 386 v1)
```

```
                            <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
                            <1>       ; 23/05/2013 - 24/05/2013 (Retro UNIX 8086 v1)
                            <1>       ; System call registers: ebx, ecx, edx (through 'sysenter')
                            <1>       ;
                            <1>       ; EBX = File descriptor
                            <1>       ;call  getf1 ; calling point in 'getf' from 'rw1'
                            <1>       ;jc    short device_rw ; read/write data from/to device
                            <1>       ; EAX = First cluster of the file
                            <1>
0000DFA6 83F802             <1>       cmp    eax, 2
0000DFA9 7217               <1>       jb     short rw2
                            <1>       ;
0000DFAB 890D[84030300]     <1>       mov    [u.base], ecx       ; buffer address/offset
                            <1>                                  ;(in the user's virtual memory space)
0000DFB1 8915[88030300]     <1>       mov    [u.count], edx
                            <1>
0000DFB7 C705[C8030300]0000- <1>      mov    dword [u.error], 0 ; reset the last error code
0000DFBF 0000               <1>
0000DFC1 C3                 <1>       retn
                            <1>
                            <1> rw2:
0000DFC2 B80A000000         <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
0000DFC7 A3[C8030300]       <1>       mov    dword [u.error], eax
0000DFCC C3                 <1>       retn
                            <1> rw3:
0000DFCD B80B000000         <1>       mov    eax, ERR_FILE_ACCESS ; permission denied !
0000DFD2 A3[C8030300]       <1>       mov    dword [u.error], eax
0000DFD7 F9                 <1>       stc
0000DFD8 C3                 <1>       retn
                            <1>
                            <1> device_read:
                            <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; cl = DEV_OPENMODE ; open mode
                            <1>       ; ch = DEV_ACCESS   ; access flags
                            <1>       ; al = DEV_DRIVER   ; device number (eax)
                            <1>
0000DFD9 F6C101             <1>       test   cl, 1 ; 1 = read, 2 = write, 3 = read&write
0000DFDC 74EF               <1>       jz     short rw3
                            <1>
0000DFDE 89C3               <1>       mov    ebx, eax
0000DFE0 66C1E302           <1>       shl    bx, 2 ; *4
                            <1>
0000DFE4 F6C580             <1>       test   ch, 80h ; bit 7, installable device driver flag
0000DFE7 7406               <1>       jz     short d_read_2 ; Kernel device
                            <1>       ; installable device
                            <1> d_read_1:
0000DFE9 FFA3[64960100]     <1>       jmp    dword [ebx+IDEV_RADDR-4]
                            <1> d_read_2:
0000DFEF FFA3[00480100]     <1>       jmp    dword [ebx+KDEV_RADDR-4]
                            <1>
                            <1> device_write:
                            <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; cl = DEV_OPENMODE ; open mode
                            <1>       ; ch = DEV_ACCESS   ; access flags
                            <1>       ; al = DEV_DRIVER   ; device number (eax)
                            <1>
0000DFF5 F6C102             <1>       test   cl, 2 ; 1 = read, 2 = write, 3 = read&write
0000DFF8 74D3               <1>       jz     short rw3
                            <1>
0000DFFA 89C3               <1>       mov    ebx, eax
0000DFFC 66C1E302           <1>       shl    bx, 2 ; *4
                            <1>
0000E000 F6C580             <1>       test   ch, 80h ; bit 7, installable device driver flag
0000E003 7406               <1>       jz     short d_write_2 ; Kernel device
                            <1>       ; installable device
                            <1> d_write_1:
0000E005 FFA3[84960100]     <1>       jmp    dword [ebx+IDEV_WADDR-4]
                            <1> d_write_2:
0000E00B FFA3[50480100]     <1>       jmp    dword [ebx+KDEV_WADDR-4]
                            <1>
                            <1>
                            <1> sysemt: ; enable (or disable) multi tasking -time sharing-
                            <1>       ;
                            <1>       ; 23/05/2016 - TRDOS 386 (TRDOS v2.0)
                            <1>       ; 14/05/2015 (Retro UNIX 386 v1)
                            <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>
0000E011 803D[B0030300]00   <1>       cmp    byte [u.uid], 0 ; root ?
                            <1>       ;ja    error
```

```
5345 0000E018 0F87D3F8FFFF        <1>       ja    badsys ; 14/05/2015
5346                              <1>       ;
5347 0000E01E FA                  <1>       cli
5348 0000E01F 881D[82950100]      <1>       mov   [multi_tasking], bl ; 0 to disable, >0 to enable
5349 0000E025 E9A2F7FFFF          <1>       jmp   sysret
5350                              <1>
5351                              <1> systimer:
5352                              <1>       ; 02/01/2017
5353                              <1>       ; 21/12/2016
5354                              <1>       ; 19/12/2016
5355                              <1>       ; 10/12/2016 (callback)
5356                              <1>       ; 10/06/2016
5357                              <1>       ; 07/06/2016
5358                              <1>       ; 06/06/2016
5359                              <1>       ; 21/05/2016
5360                              <1>       ; 19/05/2016
5361                              <1>       ; 18/05/2016 - TRDOS 386 (TRDOS v2.0)
5362                              <1>       ; (TRDOS 386 feature only!)
5363                              <1>       ;
5364                              <1>       ; (start or stop timer event(s))
5365                              <1>       ;
5366                              <1>       ; INPUT ->
5367                              <1>       ;     BL = Signal return byte (response byte)
5368                              <1>       ;          (Any requested value between 0 and 255)
5369                              <1>       ;          (Kernel will put it at the requested address)
5370                              <1>       ;     BH = Time count unit
5371                              <1>       ;          0 = Stop timer event
5372                              <1>       ;          1 = 18.2 ticks per second
5373                              <1>       ;          2 = 10 milliseconds
5374                              <1>       ;          3 = 1 second (for real time clock interrupt)
5375                              <1>       ;          4 = time/tick count in current time count unit
5376                              <1>       ;         // 10/12/2016
5377                              <1>       ;         80h = Stop timer event (callback method)
5378                              <1>       ;         81h = 18.2 ticks per second, callback method
5379                              <1>       ;         82h = 10 milliseconds, callback method
5380                              <1>       ;         83h = 1 second (for RTC int), callback method
5381                              <1>       ;         84h = current time count unit, callback method
5382                              <1>       ;
5383                              <1>       ;         Note: Only 03h or 83h will set real time clock
5384                              <1>       ;              (RTC) events (Others are for PIT events)!
5385                              <1>       ;
5386                              <1>       ;     NOTE: If callback (user service) method is used,
5387                              <1>       ;         EDX will point to the return address (of service
5388                              <1>       ;         procedure) in user's space instead of signal
5389                              <1>       ;         response byte address. (TRDOS 386 kernel will
5390                              <1>       ;         direct the cpu to that address -in user's space-
5391                              <1>       ;         at the return of system call or interrupt
5392                              <1>       ;         just after the adjusted count/time is elapsed.)
5393                              <1>       ;         User's sevice routine must be ended with a
5394                              <1>       ;         'iret'. Normal return addresses from system
5395                              <1>       ;         calls or and interrupts will be kept same except
5396                              <1>       ;         the timer returns.
5397                              <1>       ;
5398                              <1>       ;     BH = 0 -> Stop timer event
5399                              <1>       ;     BL = Timer event number (1 to 255) if BH = 0
5400                              <1>       ;          If BL = 0, all timer events (which are belongs
5401                              <1>       ;           to running process) will be stopped
5402                              <1>       ;     ECX = Time/Tick count (depending on time count unit)
5403                              <1>       ;     EDX = Signal return (Response) byte address
5404                              <1>       ;           (virtual address in user's memory space)
5405                              <1>       ; OUTPUT ->
5406                              <1>       ;     AL = Timer event number   (1 to 255) (max. value = 16)
5407                              <1>       ;     IF BH Input = 0 & CF = 0 & AL = 0 ->
5408                              <1>       ;          timer event(s) has/have been stopped/finished
5409                              <1>       ;     CF = 1 & AL = 0 -> no timer setting space to set
5410                              <1>       ;     CF = 1 & AL > 0 -> timer count unit is not usable
5411                              <1>       ;
5412                              <1>       ;     NOTE: To modify a time count for a user function,
5413                              <1>       ;           at first, current timer event must be stopped
5414                              <1>       ;           then a new timer event (which is related with
5415                              <1>       ;           same user function) must be started.
5416                              <1>       ;
5417                              <1>       ;           Signal return (response) byte may be used for
5418                              <1>       ;           several purposes. Kernel will put this value
5419                              <1>       ;           to requested address during timer interrupt,
5420                              <1>       ;           program/user can check this value to understand
5421                              <1>       ;           which event has been occurred and what is changed.
5422                              <1>       ;           (Multi timer events can share same signal address)
5423                              <1>       ;
5424                              <1>       ;     NOTE: If the process is running while the time count
5425                              <1>       ;           is reached, kernel will put signal return (response)
5426                              <1>       ;           byte value at requested address during timer
5427                              <1>       ;           interrupt and the process will continue to run.
5428                              <1>       ;           Program/process must call (jump to) it's timer event
5429                              <1>       ;           function as required, for checking the timer event
5430                              <1>       ;           status via signal return (response) byte address.
5431                              <1>       ;
5432                              <1>       ;           If the process is not running (waiting or sleeping
5433                              <1>       ;           or released) while the time count is reached,
5434                              <1>       ;           it is restarted from where it left, to ensure
5435                              <1>       ;           proper multi media (video, audio, clock, timer)
5436                              <1>       ;           functionality.
5437                              <1>       ;
5438                              <1>       ;           (It is better to use 'syswait' or 'syssleep',
5439                              <1>       ;           or 'sysrele' system call just after the timer
5440                              <1>       ;           function. Otherwise, timer events may block other
5441                              <1>       ;           processes which are not using timer events.)

5442                              <1>       ;
5443                              <1>       ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
5444                              <1>       ;      Owner:             resb 1 ; 0 = free
5445                              <1>       ;                        ;>0 = process number (u.uno)
5446                              <1>       ;     Calback:     resb 1 ; 1 = callback, 0 = response byte
5447                              <1>       ;     Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
5448                              <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)
0000E02A C605[C09A0100]00       <1>       mov    byte [tcallback], 0
0000E031 C605[C19A0100]00       <1>       mov    byte [trtc], 0
0000E038 C705[D0030300]0000-    <1>       mov    dword [u.tcb], 0 ; this is not necessary...
0000E040 0000                   <1>
                                <1>
0000E042 80FF80                 <1>       cmp    bh, 80h
0000E045 7225                   <1>       jb     short systimer_cb2
0000E047 7704                   <1>       ja     short systimer_cb0
                                <1>
0000E049 31D2                   <1>       xor    edx, edx ; 0, reset callback address
0000E04B EB0B                   <1>       jmp    short systimer_cb1
                                <1>
                                <1> systimer_cb0:
0000E04D 80FF84                 <1>       cmp    bh, 84h
0000E050 7764                   <1>       ja     short systimer_5 ;  undefined, error
                                <1>
                                <1>       ;mov   byte [tcallback], 1 ; 19/12/2016
0000E052 FE05[C09A0100]         <1>       inc    byte [tcallback]
                                <1>
                                <1> systimer_cb1:
0000E058 0FB635[B3030300]       <1>       movzx  esi, byte [u.uno] ; process number
0000E05F 66C1E602               <1>       shl    si, 2
0000E063 8996[0C010300]         <1>       mov    [esi+p.tcb-4], edx ; set process timer callback address
                                <1>                                 ; (overwrite prev value if it is set!)
0000E069 80E77F                 <1>       and    bh, 7Fh
                                <1>
                                <1> systimer_cb2:
0000E06C 80FF02                 <1>       cmp    bh, 2
0000E06F 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)
0000E071 774B                   <1>       ja     short systimer_6
                                <1>
0000E073 20FF                   <1>       and    bh, bh
0000E075 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:
0000E07B B00A                   <1>       mov    al, 10 ; (*)
                                <1>
                                <1> systimer_0:
0000E07D B710                   <1>       mov    bh, 16
                                <1>       ;
0000E07F 383D[83950100]         <1>       cmp    [timer_events], bh ; 16 ; 07/06/2016
0000E085 7319                   <1>       jnb    short systimer_3  ; max. 16 timer events
                                <1>       ;
0000E087 50                     <1>       push   eax ; (*)
                                <1>
0000E088 BF[60040300]           <1>       mov    edi, timer_set  ; beginning address of timer events
                                <1>                              ; setting space
0000E08D 30C0                   <1>       xor    al, al ; 0
                                <1> systimer_1:
0000E08F FEC0                   <1>       inc    al
0000E091 803F00                 <1>       cmp    byte [edi], 0       ; is it free space ?
0000E094 7639                   <1>       jna    short systimer_7 ; yes
0000E096 FECF                   <1>       dec    bh
0000E098 7405                   <1>       jz     short systimer_2
0000E09A 83C710                 <1>       add    edi, 16
0000E09D EBF0                   <1>       jmp    short  systimer_1 ; next event space
                                <1>
                                <1> systimer_2:
0000E09F 58                     <1>       pop    eax ; (*) discard
                                <1> systimer_3:
0000E0A0 C605[64030300]00       <1>       mov    byte [u.r0], 0
                                <1>
                                <1> systimer_4:
0000E0A7 C705[C8030300]1B00-    <1>       mov    dword [u.error], ERR_MISC
0000E0AF 0000                   <1>
                                <1>                                    ; one of miscellaneous/other errors
0000E0B1 E9F6F6FFFF             <1>       jmp    error ; cf -> 1
                                <1>
                                <1> systimer_5:
0000E0B6 883D[64030300]         <1>       mov    [u.r0], bh ; Time count unit (=2 or >3)
0000E0BC EBE9                   <1>       jmp    short systimer_4 ; 07/06/2016
                                <1>
                                <1> systimer_6:
0000E0BE 80FF04                 <1>       cmp    bh, 4
0000E0C1 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
0000E0C3 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>
0000E0C5 B0B6                   <1>       mov    al, 182 ; (*) ; 18.2 * 10
0000E0C7 FE05[C19A0100]         <1>       inc    byte [trtc] ; timer event via real time clock
0000E0CD EBAE                   <1>       jmp    short systimer_0
                                <1>
                                <1> systimer_7:
0000E0CF A2[64030300]           <1>       mov    [u.r0], al ; timer event number
```

```
                                <1>       ;
                                <1>       ; edi = address of empty timer event area
0000E0D4 A0[B3030300]           <1>       mov    al, [u.uno]
0000E0D9 FA                     <1>       cli    ; disable interrupts
0000E0DA AA                     <1>       stosb  ; process number
0000E0DB A0[C09A0100]           <1>       mov    al, [tcallback] ; timer callback flag
0000E0E0 AA                     <1>       stosb  ; 1= callback method, 0= signal response byte method
0000E0E1 A0[C19A0100]           <1>       mov    al, [trtc] ; timer interrupt type
0000E0E6 AA                     <1>       stosb  ; 1= real time clock, 0= programmable interval timer
0000E0E7 88D8                   <1>       mov    al, bl ; Signal return (Response) value
0000E0E9 AA                     <1>       stosb   ; response byte
0000E0EA 58                     <1>       pop    eax ; (*) ; 10 or 182
0000E0EB 89D3                   <1>       mov    ebx, edx ; virtual address for response/signal byte
0000E0ED F7E1                   <1>       mul    ecx
                                <1>       ; (eax = 10 * count of 18.2 Hz timer ticks)
                                <1>       ; (count down step = 10)
0000E0EF AB                     <1>       stosd  ; count limit (reset value)
0000E0F0 AB                     <1>       stosd  ; current count value
                                <1>
                                <1>       ; 19/12/2016
0000E0F1 803D[C09A0100]00       <1>       cmp    byte [tcallback], 0 ; timer callback method ?
0000E0F8 7604                   <1>       jna    short systimer_17 ; no
0000E0FA 89D8                   <1>       mov    eax, ebx ; virtual address for callback routine
0000E0FC 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
0000E0FE FE05[C29A0100]         <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.
0000E104 E8B380FFFF             <1>       call   get_physical_addr
0000E109 721A                   <1>       jc     short systimer_8 ; 07/06/2016
                                <1>       ; eax = physical address of the virtual address in user's space
                                <1> systimer_18:
0000E10B AB                     <1>       stosd  ; response addr (physical) or callback addr (virtual)
0000E10C FE05[83950100]         <1>       inc    byte [timer_events] ; 07/06/201
                                <1>       ; 02/01/2017
0000E112 0FB605[B3030300]       <1>       movzx  eax, byte [u.uno]
0000E119 FE80[FF000300]         <1>       inc    byte [eax+p.timer-1]
                                <1>       ;
0000E11F FB                     <1>       sti    ; enable interrupts
0000E120 E9A7F6FFFF             <1>       jmp    sysret
                                <1>
                                <1> systimer_8:
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
0000E125 28C0                   <1>       sub    al, al ; 0
0000E127 8847F4                 <1>       mov    [edi-12], al ; clear process number (free timer event)
                                <1>       ;mov   dword [edi], eax ; 0
0000E12A FB                     <1>       sti
0000E12B A2[64030300]           <1>       mov    [u.r0], al ; 0
0000E130 E977F6FFFF             <1>       jmp    error
                                <1>
                                <1> systimer_9:
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
0000E135 28C0                   <1>       sub    al, al
0000E137 A2[64030300]           <1>       mov    byte [u.r0], al ; 0
0000E13C 3805[83950100]         <1>       cmp     byte [timer_events], al ;  0
0000E142 7631                   <1>       jna    short systimer_12
                                <1>
                                <1>       ; Note: ecx and edx are undefined here
                                <1>       ;      (for stop timer function)
                                <1>
0000E144 BE[60040300]           <1>       mov    esi, timer_set  ; beginning address of timer events
                                <1>                               ; setting space
0000E149 A0[B3030300]           <1>       mov    al, [u.uno]
                                <1>
0000E14E B710                   <1>       mov    bh, 16
                                <1>
0000E150 08DB                   <1>       or     bl, bl
0000E152 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)
0000E154 FA                     <1>       cli    ; disable interrupts
                                <1> systimer_10:
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
0000E155 8A26                   <1>       mov    ah, [esi]
0000E157 08E4                   <1>       or     ah, ah ; 0 ?
0000E159 7411                   <1>       jz     short systimer_11
0000E15B 38C4                   <1>       cmp    ah, al ; is the process number (owner) same ?
0000E15D 750D                   <1>       jne    short systimer_11 ; no
                                <1>
                                <1>       ;mov   byte [esi], 0
0000E15F 66C7060000             <1>       mov    word [esi], 0 ; clear
                                <1>       ;mov   dword [esi+12], 0 ; clear
                                <1>
0000E164 FE0D[83950100]         <1>       dec    byte [timer_events]
0000E16A 7409                   <1>       jz     short systimer_12
                                <1>
                                <1> systimer_11:
0000E16C FECF                   <1>       dec    bh
0000E16E 7405                   <1>       jz     short systimer_12
0000E170 83C610                 <1>       add    esi, 16
0000E173 EBE0                   <1>       jmp    short systimer_10
                                <1>
                                <1> systimer_12:
0000E175 0FB635[B3030300]       <1>       movzx  esi, byte [u.uno]
0000E17C 08DB                   <1>       or     bl, bl ; all timer events or one timer event ?
0000E17E 740C                   <1>       jz     short systimer_13
```

```
5657 0000E180 8A9E[FF000300]      <1>       mov   bl, [esi+p.timer-1]
5658 0000E186 20DB                <1>       and   bl, bl ; previous number of timer events for the process
5659 0000E188 7408                <1>       jz    short systimer_14
5660 0000E18A FECB                <1>       dec   bl  ; previous number of timer events for the process - 1
5661                              <1> systimer_13:
5662 0000E18C 889E[FF000300]      <1>       mov   [esi+p.timer-1], bl ; 0 ; no timer events for process
5663                              <1> systimer_14:
5664 0000E192 FB                  <1>       sti   ; enable interrupts
5665 0000E193 E934F6FFFF          <1>       jmp   sysret
5666                              <1>
5667                              <1> systimer_15:
5668 0000E198 38FB                <1>       cmp   bl, bh ; 16
5669 0000E19A 0F8707FFFFFF        <1>       ja    systimer_4      ; max. 16 timer events !
5670                              <1>       ;
5671 0000E1A0 88DA                <1>       mov   dl, bl
5672 0000E1A2 FECA                <1>       dec   dl  ; 16 -> 15 ... 1 -> 0
5673 0000E1A4 C0E204              <1>       shl   dl, 4 ; * 16
5674 0000E1A7 0FB6FA              <1>       movzx edi, dl
5675 0000E1AA 01F7                <1>       add   edi, esi ; timer_set
5676                              <1>
5677 0000E1AC 3A07                <1>       cmp   al, [edi] ; process number
5678 0000E1AE 0F85F3FEFFFF        <1>       jne   systimer_4
5679                              <1>
5680                              <1>       ; same process ID
5681 0000E1B4 FA                  <1>       cli   ; disable interrupts
5682                              <1>       ; 10/06/2016 ; 02/01/2017
5683                              <1>       ;mov  byte [edi], 0
5684 0000E1B5 66C7070000          <1>       mov   word [edi], 0 ; clear
5685                              <1>       ;mov  dword [edi+12], 0 ; clear
5686 0000E1BA FE0D[83950100]      <1>       dec   byte [timer_events]
5687 0000E1C0 EBB3                <1>       jmp   short systimer_12
5688                              <1>
5689                              <1> sysvideo: ; VIDEO DATA TRANSFER FUNCTIONS
5690                              <1>       ; 06/03/2021
5691                              <1>       ; 02/03/2021
5692                              <1>       ; 28/02/2021
5693                              <1>       ; 27/02/2021
5694                              <1>       ; 26/02/2021
5695                              <1>       ; 25/02/2021
5696                              <1>       ; 21/02/2021, 22/02/2021, 23/02/2021
5697                              <1>       ; 15/02/2021, 16/02/2021, 18/02/2021
5698                              <1>       ; 10/02/2021, 11/02/2021, 12/02/2021
5699                              <1>       ; 07/02/2021, 08/02/2021
5700                              <1>       ; 01/02/2021, 02/02/2021, 05/02/2021
5701                              <1>       ; 29/01/2021, 30/01/2021, 31/01/2021
5702                              <1>       ; 23/01/2021, 24/01/2021, 28/01/2021
5703                              <1>       ; 18/01/2021, 19/01/2021, 22/01/2021
5704                              <1>       ; 04/01/2021, 10/01/2021, 11/01/2021
5705                              <1>       ; 01/01/2021, 02/01/2021, 03/01/2021
5706                              <1>       ; 28/12/2020, 29/12/2020, 30/12/2020
5707                              <1>       ; 25/12/2020, 26/12/2020
5708                              <1>       ; 21/12/2020, 23/12/2020
5709                              <1>       ; 12/12/2020, 14/12/2020
5710                              <1>       ; 10/12/2020, 11/12/2020
5711                              <1>       ; 03/12/2020, 04/12/2020
5712                              <1>       ; 22/11/2020, 23/11/2020
5713                              <1>       ; 21/11/2020 (TRDOS 386 v2.0.3)
5714                              <1>       ; 12/05/2017
5715                              <1>       ; 11/07/2016
5716                              <1>       ; 13/06/2016
5717                              <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
5718                              <1>       ;
5719                              <1>       ; VIDEO DATA TRANSFER FUNCTIONS:
5720                              <1>       ;
5721                              <1>       ; Inputs:
5722                              <1>       ;            ; 07/02/2021
5723                              <1>       ;     BH = 0 = VIDEO BIOS Mode 3, tty/text mode data transfers
5724                              <1>       ;          BL =
5725                              <1>       ;            Bits 0&1, Transfer direction
5726                              <1>       ;                  0 - System to system
5727                              <1>       ;                  1 - User to system
5728                              <1>       ;                  2 - System to user
5729                              <1>       ;                  3 - Exhange (Swap) - 28/01/2021
5730                              <1>       ;            Bits 2, Transfer Type
5731                              <1>       ;                  0 - Display page (complete) transfer
5732                              <1>       ;                  1 - Display page window (col,row) transfer
5733                              <1>       ;            ; 28/01/2021
5734                              <1>       ;            Bits 3..7 - Reserved, undefined (must be 0)
5735                              <1>       ;             ; 28/01/2021
5736                              <1>       ;           /// BL = 0 -> System to system (display page) transfer
5737                              <1>       ;              CL = Source page (0FFh = current video page)
5738                              <1>       ;              DL = Destination page (0FFh = current video page)
5739                              <1>       ;              (Note: Nothing to do if src & dest are same page)
5740                              <1>       ;           /// BL = 1&2 -> user to system & system to user transfer
5741                              <1>       ;              ECX = User's buffer address
5742                              <1>       ;              DL = Video page (0FFh = current video page)
5743                              <1>       ;           /// BL = 3 -> exchange (swap) display page ; 28/01/2021
5744                              <1>       ;              ECX = User's buffer address
5745                              <1>       ;              DL = Video page (0FFh = current video page)
5746                              <1>       ;              EDI = Swap address in user's memory (must be > 0)
5747                              <1>       ;           /// BL = 5&6&7 -> user to system, system to user transfer
5748                              <1>       ;            (system window is in current/active display page)
5749                              <1>       ;              ESI = User's buffer address
5750                              <1>       ;              ECX Low 16 bits = Top left column (X1 position)
5751                              <1>       ;              ECX High 16 bits = Top row (Y1 position)
5752                              <1>       ;              EDX Low 16 bits = Bottom right column (X2 position)
5753                              <1>       ;              EDX High 16 bits = Bottom row (Y2 position)
5754                              <1>       ;              If BL = 5 or BL bit 0 & bit 1 are 1 ; 28/01/2021
5755                              <1>       ;              EDI = Swap address (in user's memory space)
5756                              <1>       ;              (If swap address > 0, previous content of the window
5757                              <1>       ;              will be saved into swap area in user's memory space)
5758                              <1>       ;           /// BL = 4 -> system to system transfer
5759                              <1>       ;              ESI = System's source buffer (video page) address
5760                              <1>       ;              ECX Low 16 bits = Top left column (X1 position)
5761                              <1>       ;              ECX High 16 bits = Top row (Y1 position)
```

```
5762                    <1>     ;               EDX Low 16 bits = Bottom right column (X2 position)
5763                    <1>     ;               EDX High 16 bits = Bottom row (Y2 position)
5764                    <1>     ;               EDI = System's destination buffer (video page) addr
5765                    <1>     ;
5766                    <1>     ;              ; 06/02/2021
5767                    <1>     ;              ; 05/02/2021
5768                    <1>     ;              ; 01/02/2021, 02/02/2021
5769                    <1>     ;              ; 30/01/2021, 31/02/2021
5770                    <1>     ;              ; 29/01/2021 (major modification)
5771                    <1>     ;              ; 23/11/2020 (major modification)
5772                    <1>     ;              ; 22/11/2020 (bugfixes and extensions)
5773                    <1>     ;       BH = 1 = VGA Graphics (0A0000h) data transfers
5774                    <1>     ;              BL bit 7
5775                    <1>     ;                 resolution (screen width) option
5776                    <1>     ;                 0 = 320 pixels
5777                    <1>     ;                 1 = 640 pixels
5778                    <1>     ;              .. followings are same with SVGA transfer function
5779                    <1>     ;              BL bit 6
5780                    <1>     ;                 direction option
5781                    <1>     ;                 0 = user to system (video memory)
5782                    <1>     ;                 1 = system to user
5783                    <1>     ;              BL bit 5
5784                    <1>     ;                 masked/direct (non-masked) operations
5785                    <1>     ;                 1 = masked, 0 = non-masked (direct)
5786                    <1>     ;              BL bit 4
5787                    <1>     ;                 page/window option
5788                    <1>     ;                 1 = window, 0 = display page (screen)
5789                    <1>     ;              BL bit 0 to 3 (pixel operation types)
5790                    <1>     ;                 0 = Copy pixels (colors) ((mask color))
5791                    <1>     ;                 1 = Change (New, Fill) color
5792                    <1>     ;                 2 = Add color
5793                    <1>     ;                 3 = Sub color
5794                    <1>     ;                 4 = OR color
5795                    <1>     ;                 5 = AND color
5796                    <1>     ;                 6 = XOR color
5797                    <1>     ;                 7 = NOT color
5798                    <1>     ;                 8 = NEG color
5799                    <1>     ;                 9 = INC color
5800                    <1>     ;                10 = DEC color
5801                    <1>     ;                11 = Mix (Average) colors
5802                    <1>     ;                12 = Replace pixel colors
5803                    <1>     ;                13 = Copy pixel block(s)
5804                    <1>     ;                14 = Write line(s)
5805                    <1>     ;                15 = Write character (font)
5806                    <1>     ;
5807                    <1>     ;           Input Registers for pixel operations:
5808                    <1>     ;               Same with LFB data transfer function below
5809                    <1>     ;
5810                    <1>     ;              ; 25/02/2021
5811                    <1>     ;              ; 05/02/2021, 06/02/2021
5812                    <1>     ;              ; 01/02/2021, 02/02/2021
5813                    <1>     ;              ; 30/01/2021, 31/02/2021
5814                    <1>     ;              ; 29/01/2021 (major modification)
5815                    <1>     ;              ; 23/11/2020 (major modification)
5816                    <1>     ;              ; 22/11/2020 (bugfixes and extensions)
5817                    <1>     ;       BH = 2 = Super VGA, LINEAR FRAME BUFFER data transfers
5818                    <1>     ;              BL bit 7
5819                    <1>     ;                 unused (invalid), must be 0
5820                    <1>     ;              BL bit 6
5821                    <1>     ;                 direction option
5822                    <1>     ;                 0 = user to system (video memory)
5823                    <1>     ;                 1 = system to user
5824                    <1>     ;              BL bit 5
5825                    <1>     ;                 masked/direct (non-masked) operations
5826                    <1>     ;                 1 = masked, 0 = non-masked (direct)
5827                    <1>     ;              BL bit 4
5828                    <1>     ;                 page/window option
5829                    <1>     ;                 1 = window, 0 = display page (screen)
5830                    <1>     ;              BL bit 0 to 3 (pixel operation types)
5831                    <1>     ;                 0 = Copy pixels (colors) ((mask color))
5832                    <1>     ;                 1 = Change (New, Fill) color
5833                    <1>     ;                 2 = Add color
5834                    <1>     ;                 3 = Sub color
5835                    <1>     ;                 4 = OR color
5836                    <1>     ;                 5 = AND color
5837                    <1>     ;                 6 = XOR color
5838                    <1>     ;                 7 = NOT color
5839                    <1>     ;                 8 = NEG color
5840                    <1>     ;                 9 = INC color
5841                    <1>     ;                10 = DEC color
5842                    <1>     ;                11 = Mix (Average) colors
5843                    <1>     ;                12 = Replace pixel colors
5844                    <1>     ;                13 = Copy pixel block(s)
5845                    <1>     ;                14 = Write line(s)
5846                    <1>     ;                15 = Write character (font)
5847                    <1>     ;
5848                    <1>     ;           Note: If HW of EBX > 0, it is VESA VBE mode number
5849                    <1>     ;                 otherwise, function will be applied
5850                    <1>     ;                 to current (VESA VBE) video mode.
5851                    <1>     ;
5852                    <1>     ;           Input Registers for pixel operations:
5853                    <1>     ;              -- user to system & system to system --
5854                    <1>     ;              -- (BL = 0 to 0Fh) -- non-masked, screen --
5855                    <1>     ;              -- (BL = 10h to 1Fh) -- non-masked, window --
5856                    <1>     ;              -- (BL = 20h to 2Fh) -- masked, screen --
5857                    <1>     ;              -- (BL = 30h to 3Fh) -- masked, window --
5858                    <1>     ;              (*) window, (**) masked (***) sys to sys
5859                    <1>     ;              for BL bit 0 to 3
5860                    <1>     ;              00h: COPY PIXELS
5861                    <1>     ;                If BL bit 4 = 0 ; 21/02/2021
5862                    <1>     ;                   full screen copy
5863                    <1>     ;                   ECX & EDX will not be used
5864                    <1>     ;                  (user buffer must fit to display page)
5865                    <1>     ;                If BL bit 4 = 1 ; 21/02/2021
5866                    <1>     ;                  ECX = start position (row, column) (*)
```

```
                        ;                   (HW = row, CX = column)
                        ;               EDX = size (rows, colums) (*)
                        ;                   (HW = rows, DX = columns)
                        ;                     (0 -> invalid)
                        ;                     (1 -> horizontal or vertical line)
                        ;               ESI = user's buffer address
                        ;               EDI = mask color (**) ; 25/02/2021
                        ;                   (this color will be excluded)
                        ;           01h: CHANGE PIXEL COLORS
                        ;           02h: ADD PIXEL COLORS
                        ;           03h: SUB PIXEL COLORS
                        ;           04h: OR PIXEL COLORS
                        ;           05h: AND PIXEL COLORS
                        ;           06h: XOR PIXEL COLORS
                        ;           0Bh: MIX PIXEL COLORS
                        ;                CL = color (8 bit, 256 colors)
                        ;               ECX = color (16 bit and true colors)
                        ;               EDX = start position (row, column) (*)
                        ;                   (HW = row, DX = column)
                        ;               ESI = size (rows, colums) (*)
                        ;                   (HW = rows, SI = columns)
                        ;               EDI = mask color (**) ; 25/02/2021
                        ;                   (this color will be excluded)
                        ;           07h: NOT PIXEL COLORS
                        ;           08h: NEG PIXEL COLORS
                        ;           09h: INC PIXEL COLORS
                        ;           0Ah: DEC PIXEL COLORS
                        ;               ECX = start position (row, column) (*)
                        ;                   (HW = row, CX = column)
                        ;               EDX = size (rows, colums) (*)
                        ;                   (HW = rows, DX = columns)
                        ;                     (0 -> invalid)
                        ;                     (1 -> horizontal or vertical line)
                        ;               EDI = mask color (**) ; 25/02/2021
                        ;                   (this color will be excluded)
                        ;           0Ch: REPLACE PIXEL COLORS
                        ;                CL = current color (8 bit, 256 colors)
                        ;               ECX = current color (16 bit and true colors)
                        ;                DL = new color (8 bit, 256 colors)
                        ;               EDX = new color (16 bit and true colors)
                        ;               ESI = start position (row, column) (*)
                        ;                   (HW = row, SI = column)
                        ;               EDI = size (rows, colums) (*)
                        ;                   (HW = rows, DI = columns)
                        ;           0Dh: COPY PIXEL BLOCK(S) -full screen-
                        ;              -If BL bit 5 is 0-
                        ;               ECX = start position (row, column) (*)
                        ;                   (HW = row, CX = column)
                        ;               EDX = size (rows, colums) (*)
                        ;                   (HW = rows, DX = columns)
                        ;                     (0 -> invalid)
                        ;                     (1 -> horizontal or vertical line)
                        ;               ESI = destination (row, column) (***)
                        ;              -If BL bit 5 is 1-
                        ;                   CL = color (8 bit, 256 colors)
                        ;                 ECX = color (16 bit and true colors)
                        ;                 EDX = count of blocks (not bytes)
                        ;                     (limit: 2048 blocks)
                        ;                 ESI = user's buffer address
                        ;                   contains 64 bits block data
                        ;                   BLOCK ADDRESS - (row, col), dword
                        ;                   (first 32 bits)
                        ;                   BLOCK SIZE - (rows, cols), dword
                        ;                   (second 32 bits)
                        ;           ; 10/02/2021
                        ;           0Eh: WRITE LINE(s) -full screen-
                        ;              -If BL bit 5 is 0-
                        ;               CL = color (8 bit, 256 colors)
                        ;              ECX = color (16 bit and true colors)
                        ;               DX = low 12 bits - size (length)
                        ;                   high 4 bits - direction or type
                        ;                     0 - Horizontal line
                        ;                     1 - Vertical line
                        ;                   > 1 - undefined, invalid
                        ;            ESI = start position (row, column)
                        ;                (HW = row, SI = column)
                        ;              -If BL bit 5 is 1-
                        ;                CL = color (8 bit, 256 colors)
                        ;               ECX = color (16 bit and true colors)
                        ;                DX = number of lines (in user buffer)
                        ;                     (limit: 2048 lines)
                        ;               ESI = user's buffer
                        ;                     contains 64 bit data for lines
                        ;                     START POINT: 32 bit (row, col)
                        ;                     LENGTH: 32 bit
                        ;                     high 16 bits - 0
                        ;                     bit 0-11 - length
                        ;                     bit 12-15 - type (length)
                        ;           0Fh: WRITE CHARACTER (FONT)
                        ;                CL = char's color (8 bit, 256 colors)
                        ;               ECX = char's color (16 bit and true colors)
                        ;                DL = Character's ASCII code
                        ;                DH bit 0 -> font height
                        ;                    0 -> 8x16 character font
                        ;                    1 -> 8x8 character font
                        ;                DH bit 1 & 2 -> scale
                        ;                    0 = 1/1 (8 pixels per char row)
                        ;                    1 = 2/1 (16 pixels per char row)
                        ;                    2 = 3/1 (24 pixels per char row)
                        ;                    3 = 4/1 (32 pixels per char row)
                        ;                DH bit 6 -> [ufont] option (1 = use [ufont])
                        ;                If DH bit 7 = 1
                        ;                  USER FONT (from user buffer)
                        ;                     DL = 0 -> 8x8 (width: 1 byte per row)
                        ;                     DL = 1 -> 8x16
```

```
5972                    <1>     ;                   DL = 2 -> 16x16 (width: 2 bytes)
5973                    <1>     ;                   DL = 3 -> 16x32
5974                    <1>     ;                   DL = 4 -> 24x24 (width: 3 bytes)
5975                    <1>     ;                   DL = 5 -> 24x48
5976                    <1>     ;                   DL = 6 -> 32x32 (width: 4 bytes)
5977                    <1>     ;                   DL = 7 -> 32x64
5978                    <1>     ;                   DL > 7 -> invalid (unused)
5979                    <1>     ;                EDI = user's font buffer address
5980                    <1>     ;                   (NOTE: byte order is as row0,row1,row2..)
5981                    <1>     ;             ESI = start position (row, column) (*)
5982                    <1>     ;                    (HW = row, SI = column)
5983                    <1>     ;
5984                    <1>     ;           -- system to user --
5985                    <1>     ;              BL (bit 0 to 7)
5986                    <1>     ;           40h: COPY PIXELS (full screen, display page)
5987                    <1>     ;             EDI = user's buffer address
5988                    <1>     ;           41h: COPY PIXELS (window)
5989                    <1>     ;             ECX = start position (row, column) (*)
5990                    <1>     ;                  (HW = row, CX = column)
5991                    <1>     ;             EDX = size (rows, colums) (*)
5992                    <1>     ;                  (HW = rows, DX = columns)
5993                    <1>     ;                    (<=1 -> horizontal or vertical line)
5994                    <1>     ;             EDI = user's buffer address
5995                    <1>     ;
5996                    <1>     ;           Example: (29/01/2021)
5997                    <1>     ;             ecx = 00400064h (start at row 64, column 100)
5998                    <1>     ;             edx = 00320048h (size: 50 rows, 72 columns)
5999                    <1>     ;                             (end at row 114, column 172)
6000                    <1>     ;             If video memory starts at 0A0000h
6001                    <1>     ;             and if resolution is 320x200 (256 colors) ..
6002                    <1>     ;             window start offset: (64*320)+100 = 20580
6003                    <1>     ;             window size: 16072 bytes (pixels)
6004                    <1>     ;             window end offset: 20580+16072 = 36652
6005                    <1>     ;                window start address: 0A0000h+564h = 0A5064h
6006                    <1>     ; Outputs:
6007                    <1>     ;     EAX = transfer/byte count
6008                    <1>     ;
6009                    <1>     ;     NOTE: If the source or destination address passes out of
6010                    <1>     ;     video pages (display memory limits), data will not be transferred
6011                    <1>     ;     and EAX will return as 0.
6012                    <1>     ;
6013                    <1>     ; 08/02/2021
6014                    <1>     ; 07/02/2021
6015                    <1>     ; 04/01/2021
6016                    <1>     ; PIXEL READ/WRITE (in current/active video mode)
6017                    <1>     ;
6018                    <1>     ;     BH = 3 = Read/Write pixel(s) -for all graphics modes-
6019                    <1>     ;          BL =
6020                    <1>     ;            0 = Read pixel
6021                    <1>     ;            1 = Write pixel
6022                    <1>     ;            2 = swap pixel colors
6023                    <1>     ;            3 = mix pixel colors
6024                    <1>     ;          29/01/2021
6025                    <1>     ;            4 = read pixels from user defined positions
6026                    <1>     ;            5 = write single color pixels to user defined positions
6027                    <1>     ;            6 = write multi color pixels to user defined positions
6028                    <1>     ;
6029                    <1>     ;           > 6 = invalid/unimplemented
6030                    <1>     ;
6031                    <1>     ;          .. for BL = 0 to 5
6032                    <1>     ;          CL = color for writing pixel(s) or
6033                    <1>     ;          ECX = color for writing pixel(s) in true color modes
6034                    <1>     ;
6035                    <1>     ;          EDX = Offset from start of video memory (0A0000h)
6036                    <1>     ;               or start of linear frame buffer
6037                    <1>     ;
6038                    <1>     ;          07/02/2021
6039                    <1>     ;          .. for BL = 4
6040                    <1>     ;          EDI = user's destination buffer address for pixel colors
6041                    <1>     ;          29/01/2021
6042                    <1>     ;          .. for BL = 4 & 5
6043                    <1>     ;          ESI = user's source buffer address for BL = 4 & 5
6044                    <1>     ;          (buffer contains dword offset positions for pixels)
6045                    <1>     ;          EDX = number of pixels
6046                    <1>     ;          .. for BL = 6
6047                    <1>     ;          ESI = user's buffer address for BL = 6
6048                    <1>     ;          (buffer contains dword offset position and dword color
6049                    <1>     ;           value for each pixel)
6050                    <1>     ;          EDX = number of pixels
6051                    <1>     ;
6052                    <1>     ; Note:
6053                    <1>     ;     Pixel read/write will be performed in current video mode.
6054                    <1>     ;     If [CRT_MODE] < 0FFh, 0A0000h will be used
6055                    <1>     ;        as video memory and limit will be 65536
6056                    <1>     ;        (new/mix pixel color will be in CL)
6057                    <1>     ;     if [CRT_MODE] = 0FFh (VESA VBE video mode)
6058                    <1>     ;        LFB base address will be used as video memory
6059                    <1>     ;        and limit will be video page size
6060                    <1>     ;        (new/mix pixel color will be in CL)
6061                    <1>     ;
6062                    <1>     ; Outputs:
6063                    <1>     ;     EAX = pixel color (according to BL and ECX input)
6064                    <1>     ;     EAX = 0 (pixel color is 0 or there is an error)
6065                    <1>     ;          (BL will return as 0FFh if there is an error)
6066                    <1>     ;     ; 29/01/2021
6067                    <1>     ;     EAX = number of pixels (for BL input = 4&5&6)
6068                    <1>     ;
6069                    <1>     ; DIRECT (STANDARD VGA/CGA) DISPLAY MEMORY ACCESS FUNCTIONS:
6070                    <1>     ;
6071                    <1>     ;     BH = 4 = CGA direct video memory (0B8000h, 32K) access
6072                    <1>     ;            Page directory & page tables of the user's
6073                    <1>     ;            program will be updated to direct access to
6074                    <1>     ;            0B8000h (32K) video (CGA, color) memory; if
6075                    <1>     ;            there is not a permission conflict or lock!
6076                    <1>     ;              (User's program/process will have permission to
```

```
6077                    <1>     ;           access locked display memory if the owner is
6078                    <1>     ;           it's parent.)
6079                    <1>     ;
6080                    <1>     ;         Screen width = 320
6081                    <1>     ;
6082                    <1>     ;     BH = 5 = VGA direct video memory (0A0000h, 64K) access
6083                    <1>     ;           Page directory & page tables of the user's
6084                    <1>     ;           program will be updated to direct access to
6085                    <1>     ;           0A0000h (64K) video (VGA) memory; if there is not
6086                    <1>     ;           a permission conflict or lock!
6087                    <1>     ;             (User's program/process will have permission to
6088                    <1>     ;           access locked display memory if the owner is
6089                    <1>     ;           it's parent.)
6090                    <1>     ;
6091                    <1>     ;         ; 23/11/2020
6092                    <1>     ;         Screen width options = 320, 640, 800
6093                    <1>     ;
6094                    <1>     ; Outputs:
6095                    <1>     ;     EAX = Display memory address for direct access
6096                    <1>     ;           0A0000h for VGA, 0B8000h for CGA
6097                    <1>     ;     (Display memory size: 32K for CGA, 64K for VGA)
6098                    <1>     ;     EAX = 0 if display page access permission has been denied.
6099                    <1>     ;           (Locked!)
6100                    <1>     ;
6101                    <1>     ; LINEAR FRAME BUFFER ACCESS FUNCTIONS:
6102                    <1>     ;
6103                    <1>     ;     BH = 6 = Linear Frame Buffer direct video memory access
6104                    <1>     ;
6105                    <1>     ;           Page directory & page tables of the user's
6106                    <1>     ;           program will be updated to direct access to
6107                    <1>     ;           the configured LFB (Linear Frame Buffer) address,
6108                    <1>     ;           if there is not a permission conflict or lock!
6109                    <1>     ;             (User's program/process will have permission to
6110                    <1>     ;           access locked display memory if the owner is
6111                    <1>     ;           it's parent.)
6112                    <1>     ;
6113                    <1>     ;        ; 10/12/2020
6114                    <1>     ;        BL = 0FFh -> Direct LFB access for current video mode
6115                    <1>     ;        BL = XXh < 0FFh -> Direct LFB access
6116                    <1>     ;                          for VESA video mode 1XXh
6117                    <1>     ;
6118                    <1>     ;           Return: EAX = Linear Frame Buffer address
6119                    <1>     ;                  (EAX = 0 -> error)
6120                    <1>     ;                If EAX > 0
6121                    <1>     ;                  EDX = Frame Buffer Size in bytes
6122                    <1>     ;                    BH = Requested Video Mode - 100h
6123                    <1>     ;                         (VESA VBE video modes)
6124                    <1>     ;                    BL = bits per pixel
6125                    <1>     ;                        8 = 256 colors, 8
6126                    <1>     ;                         16 = 65536 colors, 5-6(G)-5
6127                    <1>     ;                         24 = RGB, 16M colors, 8-8-8
6128                    <1>     ;                         32 = RGB + alpha bytes, 8-8-8-8
6129                    <1>     ;                If BH = 0FFh
6130                    <1>     ;                   BL = VGA/CGA video mode (also EAX = 0)
6131                    <1>     ;
6132                    <1>     ;           ** Function will return with EAX = 0 if the mode
6133                    <1>     ;           is not a valid VESA VBE video mode as 1??h **
6134                    <1>     ;
6135                    <1>     ;           ECX = Pixel resolution
6136                    <1>     ;                 CX = Width (320, 640, 800, 1024, 1366, 1920)
6137                    <1>     ;                 High 16 bits of ECX = Height
6138                    <1>     ;
6139                    <1>     ;     23/11/2020
6140                    <1>     ; ***  GET VIDEO MODE & LINEAR FRAME BUFFER INFO
6141                    <1>     ;     (This function -7- also is used for VGA and CGA modes)
6142                    <1>     ;
6143                    <1>     ;     BH = 7 = Get Linear Frame Buffer info
6144                    <1>     ;
6145                    <1>     ;        ; 22/01/2021
6146                    <1>     ;        ; 10/12/2020
6147                    <1>     ;        BL = any -not used- (22/01/2021)
6148                    <1>     ;
6149                    <1>     ;           Return:
6150                    <1>     ;           EAX = Frame Buffer Address (0 = is not in use)
6151                    <1>     ;           EDX = Frame Buffer Size in bytes
6152                    <1>     ;            BH = Current Video Mode - 100h  ; 22/01/2021
6153                    <1>     ;                (VESA VBE video modes)
6154                    <1>     ;            BL = bits per pixel
6155                    <1>     ;                 8 = 256 colors, 8
6156                    <1>     ;                 16 = 65536 colors, 5-6(G)-5
6157                    <1>     ;                 24 = RGB, 16M colors, 8-8-8
6158                    <1>     ;                 32 = RGB + alpha bytes, 8-8-8-8
6159                    <1>     ;           If BH = 0FFh
6160                    <1>     ;              BL = VGA/CGA video mode (also EAX = 0)
6161                    <1>     ;
6162                    <1>     ;           Note:
6163                    <1>     ;           Alpha byte will be used as virtual color index.
6164                    <1>     ;           (32 bit pixel colors.. byte 0,1,2 rgb and 3 alpha)
6165                    <1>     ;
6166                    <1>     ;           ** Function will return with EAX = 0 if the mode
6167                    <1>     ;           is not a valid VESA VBE video mode as 1??h **
6168                    <1>     ;
6169                    <1>     ;           ECX = Pixel resolution
6170                    <1>     ;                 CX = Width (320, 640, 800, 1024, 1366, 1920)
6171                    <1>     ;                 High 16 bits of ECX = Height
6172                    <1>     ;
6173                    <1>     ;     NOTE: Each process will have it's own frame buffer
6174                    <1>     ;           address and resolution parameters in 'u' area.
6175                    <1>     ;           Then, if the current frame buffer & resolution
6176                    <1>     ;           is different, frame buffer r/w functions
6177                    <1>     ;           will use scale factor to convert process's
6178                    <1>     ;            pixel coordinates to actual screen coordinates.
6179                    <1>     ;           resolution -> dimensional scale
6180                    <1>     ;           color size -> color scale
6181                    <1>     ;          * RGB (TRUE) colors to 256 colors conversion:
```

```
6182                    <1>     ;           TRUE Colors -> 8,8,8 (R,G,B; byte 0 is R)
6183                    <1>     ;         256 colors -> 2,2,2,2 (R,G,B,L; bit 0&1 is R)
6184                    <1>     ;            bit 6&7 -> luminosity base level (0,1,2,3)
6185                    <1>     ;            bit 4&5 -> blue level (0,1,2,3)
6186                    <1>     ;            bit 2%3 -> green level (0,1,2,3)
6187                    <1>     ;            bit 0&1 -> red level (0,1,2,3)
6188                    <1>     ;         Example: total red level : luminosity + red level
6189                    <1>     ;         Luminosity base level: 0 -> 16
6190                    <1>     ;                                1 -> 32
6191                    <1>     ;                                2 -> 64
6192                    <1>     ;                                3 -> 128
6193                    <1>     ;          Color level:
6194                    <1>     ;                               0 -> 0
6195                    <1>     ;                               1 -> luminosity level
6196                    <1>     ;                               2 -> luminosity level + 64
6197                    <1>     ;                               3 -> 255
6198                    <1>     ;        Luminosity base level = min (R,G,B)
6199                    <1>     ;                if it is <16, it will be set to 16
6200                    <1>     ;        Color levels: Color values are fixed to (nearest)
6201                    <1>     ;             one of all possible set level (step) values
6202                    <1>     ;             (according to luminosity base level); then
6203                    <1>     ;             color levels are set to R-L, G-L, B-L.
6204                    <1>     ;      For example: If luminosity base level is 32
6205                    <1>     ;             all possible set values are 0, 32, 96, 255.
6206                    <1>     ;
6207                    <1>     ;        * RGB (TRUE) colors to 16 colors conversion:
6208                    <1>     ;        16 colors: R,B,G,L bits (4 bits)
6209                    <1>     ;             If any one of R,G,B >= 128 L = 1
6210                    <1>     ;             If max. value of (R,G,B) >= 32, it is 1
6211                    <1>     ;                else all color bits (R&G&B&L) are 0
6212                    <1>     ;             If the second value >= max. value / 2
6213                    <1>     ;                it is 1
6214                    <1>     ;             If third value value >= max. value / 2
6215                    <1>     ;                it is 1
6216                    <1>     ;         Example: R = 132, G = 64, B = 78
6217                    <1>     ;                L = 1, R = 1
6218                    <1>     ;                G < 66 --> G = 0
6219                    <1>     ;                B >= 66 --> B = 1
6220                    <1>     ;
6221                    <1>     ; 10/12/2020
6222                    <1>     ; SET VIDEO MODE (& RETURN LFB INFO for VESA VBE VIDEO MODES)
6223                    <1>     ;
6224                    <1>     ;     BH = 8 = Set Video Mode
6225                    <1>     ;
6226                    <1>     ;            BL = Requested Video Mode (method)
6227                    <1>     ;            If BL = 0FFh
6228                    <1>     ;               CX = VESA VBE Video Mode
6229                    <1>     ;               ; 11/12/2020
6230                    <1>     ;               If EDX > 0 -> LFB INFO (user) buffer addr
6231                    <1>     ;            If BL < 0FFh, it is VGA/CGA video mode and
6232                    <1>     ;                  CX & EDX will not be used
6233                    <1>     ;
6234                    <1>     ;         NOTE: The last VESA VBE video mode is 11Bh but
6235                    <1>     ;         TRDOS 386 will permit to set video mode upto 11Fh.
6236                    <1>     ;         Above 11Fh, from 140h to 1FEh, it will be accepted
6237                    <1>     ;         as Bochs/Plex86 emulator video mode and it will be
6238                    <1>     ;         used only if [vbe3] = 2 and detected
6239                    <1>     ;         video bios is BOCHS/PLEX86/QEMU/VIRTUALBOX vbios.
6240                    <1>     ;
6241                    <1>     ; Outputs:
6242                    <1>     ;     EAX = Requested (Proper) video mode number + 1
6243                    <1>     ;           ("dec eax" by user will give requested video mode),
6244                    <1>     ;
6245                    <1>     ;     If BL input is 0FFh
6246                    <1>     ;        EDX = LFBINFO table/structure (in user's buffer addr)
6247                    <1>     ;        EDX = 0 -> Invalid LFBINFO (do not use it)
6248                    <1>     ;
6249                    <1>     ;     EAX = 0 -> Error (but EDX will not be changed)
6250                    <1>     ;
6251                    <1>     ; 03/12/2020
6252                    <1>     ; VESA VBE3 VIDEO BIOS (32 bit) PROTECTED MODE INTERFACE SETTINGS
6253                    <1>     ;
6254                    <1>     ;     BH = 9 = set/get VBE3 Protected Mode Interface parameters
6255                    <1>     ;
6256                    <1>     ;            BL = 0 - Disable protected mode interface
6257                    <1>     ;                  ([pmi32] = 0)
6258                    <1>     ;                  Return: AL = 1
6259                    <1>     ;            BL = 1 - Enable protected mode Interface
6260                    <1>     ;                  ([pmi32] = 1)
6261                    <1>     ;                  Return: AL = 2
6262                    <1>     ;            BL = 2 - Get protected mode interface status
6263                    <1>     ;                  Return: AL = [pmi32] + 1 (AL = 1 or 2)
6264                    <1>     ;
6265                    <1>     ;            If [vbe3] <> 3 --> AL = 0
6266                    <1>     ;
6267                    <1>     ;            ; 17/01/2021
6268                    <1>     ;            BL = 3 - Disable/Cancel restore permission to user
6269                    <1>     ;                  Return: AL = 1 (if disabled) or 0
6270                    <1>     ;            BL = 4 - Enable/Give restore permission to user
6271                    <1>     ;                  Return: AL = 2 (if enabled) or 0
6272                    <1>     ;            BL = 5 - Get video state save/restore status
6273                    <1>     ;                   (permission status)
6274                    <1>     ;                   Return: AL = Status (enabled = 1)
6275                    <1>     ;                        ; 22/01/2021
6276                    <1>     ;                        AH = state options ([srvso])
6277                    <1>     ;            BL = 6 - Return VESA VBE number/status
6278                    <1>     ;                   Return: AX = status
6279                    <1>     ;                       if AH =  2, AL > 0 : Emulator
6280                    <1>     ;                        AH =  3, VESA VBE3 video bios
6281                    <1>     ;            ; 28/02/2021
6282                    <1>     ;            BL = 7 - Set true color mode as 32bpp (default)
6283                    <1>     ;                  Return: AX = 32 (if VBE3)
6284                    <1>     ;            NOTE: Initial/default value is 32bpp for vbe3.
6285                    <1>     ;                  Return: AX = 0 -> error
6286                    <1>     ;            BL = 8 - Set true color mode as 24bpp (default)
```

```
6287                    <1>     ;                   Return: AX = 24
6288                    <1>     ;                   ;Return: AX = 0 -> error
6289                    <1>     ;             BL = 9 - Return default/current true color mode
6290                    <1>     ;                   Return: AX = 32 (32 bpp)
6291                    <1>     ;                   Return: AX = 24 (24 bpp)
6292                    <1>     ;                   Return: AX = 0 -> error (not VESA bios)
6293                    <1>     ;
6294                    <1>     ;             BL > 9 : not implemented (28/02/2021)
6295                    <1>     ;
6296                    <1>     ;     ; 19/01/2021 ([u.uid] check)
6297                    <1>     ;     Note: Enabling/Disabling are done by root ([u.uid] = 0)
6298                    <1>     ;           while [multi_tasking] = 0.
6299                    <1>     ;
6300                    <1>     ; 23/12/2020
6301                    <1>     ; VIDEO MEMORY MAPPING:
6302                    <1>     ;     BH = 10 = Map video memory to user's buffer
6303                    <1>     ;
6304                    <1>     ;        BL = 0 : CGA memory (0B8000h) map (32K)
6305                    <1>     ;        BL = 1 : VGA memory (0A0000h) map (64K)
6306                    <1>     ;        BL = 2 : SVGA memory (LFB) map to user's buffer
6307                    <1>     ;
6308                    <1>     ;     ECX = User's buffer addr (low 12 bits will be cleared)
6309                    <1>     ;     EDX = Buffer size in bytes (if BL = 2)
6310                    <1>     ;           (will be trimmed if LFB size < EDX)
6311                    <1>     ; Return:
6312                    <1>     ;     EAX = physical address of video memory (buffer)
6313                    <1>     ;     EBX = mapped (actual) size of video memory (bytes)
6314                    <1>     ;     ECX = virtual start address of user's video buffer
6315                    <1>     ;     EDX is same with EDX input
6316                    <1>     ;
6317                    <1>     ;     (Note: Memory page boundaries will be applied
6318                    <1>     ;      to buffer size and buff start addr by rounding down.
6319                    <1>     ;      Rounded size & address values must not be zero.)
6320                    <1>     ;     -Normally, it is expected to request mapping by using
6321                    <1>     ;      correct buffer size of current or desired video mode-
6322                    <1>     ;
6323                    <1>     ;     EAX = 0 -> error ! memory can not mapped to user
6324                    <1>     ;
6325                    <1>     ; 04/01/2021
6326                    <1>     ; SET/READ COLOR PALETTE (set/read DAC color registers)
6327                    <1>     ;     ((256 colors (8bpp) VGA/CGA video hardware feature))
6328                    <1>     ;
6329                    <1>     ;     BH = 11 = Set/Read DAC color registers
6330                    <1>     ;
6331                    <1>     ;        (BL<4 Original method for std VGA video hardware)
6332                    <1>     ;        BL = 0 : Read all DAC color registers (256 colors)
6333                    <1>     ;            (6 bit colors, in RGB order)
6334                    <1>     ;        BL = 1 : Set all DAC color registers (256 colors)
6335                    <1>     ;            (6 bit colors, in RGB order)
6336                    <1>     ;        BL = 2 : Read single DAC color register
6337                    <1>     ;            (6 bit color, in RGB order)
6338                    <1>     ;           ((EAX will return with color value))
6339                    <1>     ;           CL = DAC color register (index)
6340                    <1>     ;        BL = 3 : Set/Write single DAC color register
6341                    <1>     ;            (6 bit color, in RGB order, bit 6&7 are 0)
6342                    <1>     ;           ECX byte 0 - DAC color register
6343                    <1>     ;           ECX byte 1 - Red (6 bit)
6344                    <1>     ;           ECX byte 2 - Green (6 bit)
6345                    <1>     ;           ECX byte 3 - Blue (6 bit)
6346                    <1>     ;        (BL>3 Alternative method for BMP files etc.)
6347                    <1>     ;        BL = 4 : Read all DAC color registers (256 colors)
6348                    <1>     ;            (8 bit colors, in BGR order, bit 0&1 is 0)
6349                    <1>     ;        BL = 5 : Set all DAC color registers (256 colors)
6350                    <1>     ;            (8 bit colors, in BGR order, bit 0&1 is 0)
6351                    <1>     ;        BL = 6 : Read single DAC color register
6352                    <1>     ;            (8 bit color, in BGR order, bit 0&1 is 0)
6353                    <1>     ;           ((EAX will return with color value))
6354                    <1>     ;           CL = DAC color register (index)
6355                    <1>     ;        BL = 7 : Set/Write single DAC color register
6356                    <1>     ;            (8 bit color, bit 0&1 are 0)
6357                    <1>     ;           ECX byte 0 - DAC color register
6358                    <1>     ;           ECX byte 1 - Blue (8 bit)
6359                    <1>     ;           ECX byte 2 - Green (8 bit)
6360                    <1>     ;           ECX byte 3 - Red (8 bit)
6361                    <1>     ;
6362                    <1>     ;       BL > 7 : invalid (not implemented)
6363                    <1>     ;
6364                    <1>     ;   if BL bit 2 is 1, 6 bit colors converted to 8 bit colors
6365                    <1>     ;      (low two bits of color bytes will be 0)
6366                    <1>     ;     ((color byte 00111111b will be converted to 11111100b))
6367                    <1>     ;     and RGB byte order will be
6368                    <1>     ;           byte 0 - Blue (low 2 bits are 0)
6369                    <1>     ;           byte 1 - Green (low 2 bits are 0)
6370                    <1>     ;           byte 2 - Red (low 2 bits are 0)
6371                    <1>     ;           byte 3 - pad (or zero byte)
6372                    <1>     ;     and 256 colors buffer size must be 256*4 = 1024 bytes
6373                    <1>     ;   if BL bit 2 is 0, 6 bit colors will be used directly
6374                    <1>     ;        (high two bits of 8 bit color bytes will be 0)
6375                    <1>     ;     ((dac color 111111b will be converted to 00111111b))
6376                    <1>     ;           byte 0 - Red (high 2 bits are 0)
6377                    <1>     ;           byte 1 - Green (high 2 bits are 0)
6378                    <1>     ;           byte 2 - Blue (high 2 bits are 0)
6379                    <1>     ;     and 256 colors buffer size must be 256*3 = 768 bytes
6380                    <1>     ;
6381                    <1>     ;     ECX = User's buffer addr (256*3 = 768 bytes) or
6382                    <1>     ;           Color
6383                    <1>     ;
6384                    <1>     ; Return:
6385                    <1>     ;     EAX = buffer size (for BL input = 0,1,4,5)
6386                    <1>     ;           or color value (for BL input = 2,3,6,7)
6387                    <1>     ;
6388                    <1>     ; 10/01/2021
6389                    <1>     ; SET/READ FONT DATA
6390                    <1>     ;
6391                    <1>     ;     BH = 12 = Set/Read Character Font Data
```

```
6392                    <1>      ;
6393                    <1>      ;        BL = 0 : Disable system font overwrite
6394                    <1>      ;        BL = 1 : Enable system font overwrite
6395                    <1>      ;        BL = 2 : Read system font 8x8
6396                    <1>      ;        BL = 3 : Read system font 8x14
6397                    <1>      ;        BL = 4 : Read system font 8x16
6398                    <1>      ;        BL = 5 : Read user defined font 8x8
6399                    <1>      ;        BL = 6 : Read user defined font 8x16
6400                    <1>      ;        BL = 7 : Write system font 8x8
6401                    <1>      ;        BL = 8 : Write system font 8x14
6402                    <1>      ;        BL = 9 : Write system font 8x16
6403                    <1>      ;        BL = 10 : Write user defined font 8x8
6404                    <1>      ;        BL = 11 : Write user defined font 8x16
6405                    <1>      ;
6406                    <1>      ;       BL > 11 : invalid (not implemented)
6407                    <1>      ;
6408                    <1>      ;     For BL = 1 to 11
6409                    <1>      ;        ECX = number of characters (<= 256)
6410                    <1>      ;        EDX = first character (ascii code in DL)
6411                    <1>      ;        ESI = user's buffer address
6412                    <1>      ;
6413                    <1>      ; Return:
6414                    <1>      ;     EAX = number of characters (ecx input)
6415                    <1>      ;     EAX = 0 -> error
6416                    <1>      ;     (EAX = 256 for BL = 0 and 1 if successful)
6417                    <1>      ;
6418                    <1>      ; Note: system font overwrite permission will be
6419                    <1>      ;       given if [multi_tasking] = 0
6420                    <1>      ;     and [u.uid] = 0 (BL = 1) ; 19/01/2021
6421                    <1>      ;       and if [ufont] bit 7 is 1 (BL = 7,8,9)
6422                    <1>      ;
6423                    <1>      ; 18/01/2021
6424                    <1>      ; SAVE/RESTORE STANDARD VGA VIDEO STATE
6425                    <1>      ;
6426                    <1>      ;     BH = 13 = Save/Restore std VGA video state
6427                    <1>      ;
6428                    <1>      ;        BL = 0 : Save VGA state (without DAC regs)
6429                    <1>      ;            Return: EAX = VideoStateID (>0)
6430                    <1>      ;                  EAX = 0 (failed!)
6431                    <1>      ;             (size: 110 bytes for TRDOS 386 v2.0.3)
6432                    <1>      ;        BL = 1 : Restore VGA state (without DAC regs)
6433                    <1>      ;            ECX = VideoStateID (to be verified)
6434                    <1>      ;              Return: EAX = Restore size (>0)
6435                    <1>      ;        BL = 2 : Save VGA state (complete)
6436                    <1>      ;            Return: EAX = VideoStateID (>0)
6437                    <1>      ;                  EAX = 0 (failed!)
6438                    <1>      ;             (size: 882 bytes for TRDOS 386 v2.0.3)
6439                    <1>      ;        BL = 3 : Restore VGA state (complete)
6440                    <1>      ;            ECX = VideoStateID (to be verified)
6441                    <1>      ;              Return: EAX = Restore size (>0)
6442                    <1>      ;
6443                    <1>      ;     * Above options are for saving
6444                    <1>      ;     *       video state to system memory
6445                    <1>      ;     *      (location: VBE3VIDEOSTATE, 2048 bytes)
6446                    <1>      ;
6447                    <1>      ;        BL = 4 : Save VGA state (without DAC regs)
6448                    <1>      ;            ECX = buffer address
6449                    <1>      ;            Return: EAX = transfer count
6450                    <1>      ;             (size: 110 bytes for TRDOS 386 v2.0.3)
6451                    <1>      ;            ECX = buffer address
6452                    <1>      ;        BL = 5 : Restore VGA state (without DAC regs)
6453                    <1>      ;            ECX = buffer address
6454                    <1>      ;              Return: EAX = transfer count
6455                    <1>      ;        BL = 6 : Save VGA state (complete)
6456                    <1>      ;            ECX = buffer address
6457                    <1>      ;            Return: EAX = transfer count
6458                    <1>      ;             (size: 882 bytes for TRDOS 386 v2.0.3)
6459                    <1>      ;        BL = 7 : Restore VGA state (complete)
6460                    <1>      ;            ECX = buffer address
6461                    <1>      ;              Return: EAX = transfer count
6462                    <1>      ;
6463                    <1>      ;     * Above options are for saving
6464                    <1>      ;     *       video state to user's buffer
6465                    <1>      ;     *      (buffer size: 110 bytes or 882 bytes)
6466                    <1>      ;
6467                    <1>      ;       BL > 7 : invalid (not implemented)
6468                    <1>      ;
6469                    <1>      ; 18/01/2021
6470                    <1>      ; SAVE/RESTORE SUPER VGA (VESA VBE 2/3) VIDEO STATE
6471                    <1>      ;
6472                    <1>      ;     BH = 14 = Save/Restore SVGA video state
6473                    <1>      ;
6474                    <1>      ;        BL = options
6475                    <1>      ;            bit 0 - Save (0) or Restore (1)
6476                    <1>      ;            bit 1 - controller hardware state
6477                    <1>      ;            bit 2 - BIOS data state
6478                    <1>      ;            bit 3 - DAC state
6479                    <1>      ;            bit 4 - (extended) Register state
6480                    <1>      ;            bit 5 - system (0) or user (1) memory
6481                    <1>      ;            bit 6 - verify without transfer
6482                    <1>      ;            bit 7 - not used (must be 0)
6483                    <1>      ;
6484                    <1>      ;          if bit 0 = 0 and bit 5 = 0
6485                    <1>      ;            Return:      EAX = VideoStateID (>0)
6486                    <1>      ;          if bit 0 = 1
6487                    <1>      ;            ECX = VideoStateID (bit 5 = 0)
6488                    <1>      ;            Return: EAX = restore (transfer) size
6489                    <1>      ;          if bit 5 = 1
6490                    <1>      ;            ECX = Buffer address
6491                    <1>      ;            Return: EAX = transfer count (size)
6492                    <1>      ;
6493                    <1>      ;          ECX = Buffer address or VideoStateID
6494                    <1>      ;
6495                    <1>      ;       BL > 127 : invalid (not implemented)
6496                    <1>      ;
```

```
6497                                    <1>     ;  Note: Required buffer size may be > 2048 bytes
6498                                    <1>     ;       (function fails when buff size > 2048 bytes)
6499                                    <1>     ;       proper option must be used
6500                                    <1>     ;
6501                                    <1>     ; 18/01/2021
6502                                    <1>     ; READ VESA EDID (EXTENDED DISPLAY IDENTIFICATION DATA)
6503                                    <1>     ;
6504                                    <1>     ;      BH = 15 = Read VESA EDID for connected monitor
6505                                    <1>     ;               (copy EDID to user)
6506                                    <1>     ;
6507                                    <1>     ;         BL = any
6508                                    <1>     ;
6509                                    <1>     ;      Input:
6510                                    <1>     ;          ECX = user's (EDID) buffer address
6511                                    <1>     ;              (buffer size: 128 bytes)
6512                                    <1>     ;      Output:
6513                                    <1>     ;          EAX = 128 (EDID size)
6514                                    <1>     ;          or EAX = 0 -> Error!
6515                                    <1>     ;             (EDID not ready or buffer addr error)
6516                                    <1>     ;
6517                                    <1>
6518                                    <1>     ; 16/05/2016
6519 0000E1C2 31C0                     <1>     xor    eax, eax
6520 0000E1C4 A3[64030300]             <1>     mov    [u.r0], eax
6521 0000E1C9 20FF                     <1>     and    bh, bh
6522 0000E1CB 0F858B020000             <1>     jnz    sysvideo_13 ; 11/07/2016
6523                                    <1>
6524                                    <1>     ;; 21/11/2020 (TRDOS 386 v2.0.3)
6525                                    <1>     ;; tty/text mode transfers are only for video mode 3
6526                                    <1>
6527                                    <1>     ; 22/11/2020
6528                                    <1>     ;cmp   byte [CRT_MODE], 3 ; 80x25 text, 16 colors
6529                                    <1>     ;jne   sysret ; invalid (nothing to do), [u.r0] = 0
6530                                    <1>
6531                                    <1>     ; 23/11/2020
6532                                    <1>     ; bit 7,6,5,4 of BL are reserved and it must be 0
6533                                    <1>     ;             for current 'sysvideo' version
6534                                    <1>     ;test  bl, 0F0h
6535                                    <1>     ;;jnz  sysret ; invalid (undefined) !
6536                                    <1>     ;; 28/01/2021
6537                                    <1>     ;jnz   short sysvideo_1_2 ; invalid (undefined) !
6538                                    <1>     ; 28/01/2021
6539 0000E1D1 80FB07                   <1>     cmp    bl, 7
6540 0000E1D4 776E                     <1>     ja     short sysvideo_1_2 ; invalid (undefined) !
6541                                    <1>
6542                                    <1>     ; Video mode 0, 80*25 text mode, CGA 16 colors
6543                                    <1>     ; [CRT_MODE] = 3
6544                                    <1>     ;mov   bh, bl
6545                                    <1>     ;shr   bh, 2 ; 4..7 -> 1, 8..11 -> 2, 12..15 -> 3
6546                                    <1>     ;; 21/11/2020
6547                                    <1>     ;;and  bh, bh
6548                                    <1>     ;jnz   sysvideo_4  ; Display page window transfer etc.
6549                                    <1>
6550                                    <1>     ; 28/01/2021
6551 0000E1D6 F6C304                   <1>     test   bl, 4 ; bit 2
6552 0000E1D9 0F85A2000000             <1>     jnz    sysvideo_4 ; Display page window transfer
6553                                    <1>
6554                                    <1>     ; Display page (complete) transfer
6555 0000E1DF 80FA07                   <1>     cmp    dl, 7
6556                                    <1>     ;jnz   sysret ; invalid (nothing to do), [u.r0] = 0
6557 0000E1E2 760A                     <1>     jna    short sysvideo_0 ; 28/01/2021
6558 0000E1E4 FEC2                     <1>     inc    dl ; 0FFh -> 0 ("use current video page")
6559 0000E1E6 755C                     <1>     jnz    short sysvideo_1_2  ; invalid
6560                                    <1>     ; dl = 0 -> use current current page
6561 0000E1E8 8A15[1E890100]           <1>     mov    dl, [ACTIVE_PAGE]
6562                                    <1> sysvideo_0:
6563                                    <1>     ; 28/01/2021
6564 0000E1EE 80FB03                   <1>     cmp    bl, 3
6565 0000E1F1 7206                     <1>     jb     short sysvideo_0_0
6566 0000E1F3 09FF                     <1>     or     edi, edi
6567 0000E1F5 744D                     <1>     jz     short sysvideo_1_2  ; invalid
6568 0000E1F7 89FE                     <1>     mov    esi, edi ; save swap/exchange buffer addr
6569                                    <1>     ; ecx = user buffer for new video page content
6570                                    <1>     ; esi = user (swap) buffer for saving current video page
6571                                    <1> sysvideo_0_0:
6572 0000E1F9 BF00800B00               <1>     mov    edi, 0B8000h
6573                                    <1>     ; dl = display page number, destination
6574 0000E1FE 66B80010                 <1>     mov    ax, 4096 ; 21/11/2020
6575 0000E202 20D2                     <1>     and    dl, dl
6576 0000E204 7408                     <1>     jz     short sysvideo_1
6577                                    <1>     ; 07/02/2021
6578 0000E206 88D6                     <1>     mov    dh, dl
6579                                    <1> sysvideo_0_1:
6580                                    <1>     ; page length = 4096 bytes (but page content is 80*25*2 bytes)
6581 0000E208 01C7                     <1>     add    edi, eax ; 21//11/2020 ([CRT_LEN] = 1000h for mode 3)
6582 0000E20A FECE                     <1>     dec    dh
6583 0000E20C 75FA                     <1>     jnz    short sysvideo_0_1
6584                                    <1> sysvideo_1:
6585 0000E20E 80E303                   <1>     and    bl, 3
6586 0000E211 7536                     <1>     jnz    short sysvideo_2 ; user to system display page transfer
6587                                    <1>     ; system to system video page (content) transfer
6588                                    <1>     ; cl = display page number, source
6589 0000E213 80F907                   <1>     cmp    cl, 7
6590                                    <1>     ;ja    sysret ; invalid (nothing to do), [u.r0] = 0
6591 0000E216 760A                     <1>     jna    short sysvideo_1_0
6592 0000E218 FEC1                     <1>     inc    cl ; 0FFh -> 0 ("use current video page")
6593 0000E21A 7528                     <1>     jnz    short sysvideo_1_2  ; invalid
6594 0000E21C 8A0D[1E890100]           <1>     mov    cl, [ACTIVE_PAGE]
6595                                    <1> sysvideo_1_0:
6596                                    <1>     ; 28/01/2021
6597 0000E222 38D1                     <1>     cmp    cl, dl
6598 0000E224 741E                     <1>     je     short sysvideo_1_2  ; same video page !
6599                                    <1>
6600                                    <1>     ; system to system video/display page transfer (mode 0)
6601                                    <1>     ; 21/11/2020
```

```
                                <1>       ;mov   esi, 0B8000h
                                <1>       ;movzx eax, cl
                                <1>       ;mov   edx, 4096 ; [CRT_LEN] = 1000h for video mode 3
                                <1>       ;mov   ecx, edx
                                <1>       ;mul   edx
                                <1>       ;add   esi, eax
                                <1>       ; 28/01/2021
                                <1>       ;movzx esi, cl
                                <1>       ;shl   si, 12 ; * 4096
                                <1>       ;add   esi, 0B8000h
                                <1>
                                <1>       ; 28/01/2021
0000E226 A3[64030300]           <1>       mov    [u.r0], eax ; 4096
0000E22B BE00800B00             <1>       mov    esi, 0B8000h
0000E230 08C9                   <1>       or     cl, cl
0000E232 740A                   <1>       jz     short sysvideo_1_1
                                <1>       ; 07/02/2021
0000E234 88C8                   <1>       mov    al, cl ; display/video page
0000E236 30E4                   <1>       xor    ah, ah
0000E238 66C1E00C               <1>       shl    ax, 12 ; * 4096
0000E23C 01C6                   <1>       add    esi, eax
                                <1> sysvideo_1_1:
                                <1>       ; 21/11/2020
                                <1>       ;;mov  ecx, 4096
                                <1>       ;mov   ecx, eax ; 4096
                                <1>       ;;mov  [u.r0], ecx ; 4096 bytes
                                <1>       ; 28/01/2021
                                <1>       ;mov   [u.r0], cx
0000E23E 31C9                   <1>       xor    ecx, ecx
0000E240 B504                   <1>       mov    ch, 4 ; mov ecx, 1024
                                <1>       ;shr   cx, 2 ; / 4
0000E242 F3A5                   <1>       rep    movsd
                                <1> sysvideo_1_2:
0000E244 E983F5FFFF             <1>       jmp    sysret
                                <1> sysvideo_2:
0000E249 80FB02                 <1>       cmp    bl, 2
                                <1>       ;ja    sysret; invalid (user to user), [u.r0] = 0
                                <1>       ; 28/01/2021
0000E24C 7226                   <1>       jb     short sysvideo_3 ; user to system
0000E24E 7404                   <1>       je     short sysvideo_2_0 ; system to user
                                <1>       ; bl = 3
0000E250 89CA                   <1>       mov    edx, ecx ; save user's buffer addr
0000E252 89F1                   <1>       mov    ecx, esi ; save swap address
                                <1> sysvideo_2_0:
                                <1>       ; bl = 2 (or bl = 3, stage 1)
                                <1>       ; system to user video/display page transfer (mode 0)
0000E254 89FE                   <1>       mov    esi, edi
0000E256 89CF                   <1>       mov    edi, ecx ; user buffer ; 28/01/2021
                                <1>       ; 21/11/2020
0000E258 89C1                   <1>       mov    ecx, eax ; 4096
0000E25A E8C5370000             <1>       call   transfer_to_user_buffer ; fast transfer
                                <1>       ;jc    sysret ; [u.r0] = 0
0000E25F 72E3                   <1>       jc     short sysvideo_1_2 ; 28/01/2021
                                <1>       ; 28/01/2021
0000E261 80FB03                 <1>       cmp    bl, 3
0000E264 7408                   <1>       je     short sysvideo_2_2
                                <1> sysvideo_2_1:
                                <1>       ; 21/11/2020
0000E266 890D[64030300]         <1>       mov    [u.r0], ecx
                                <1>       ;;mov  [u.r0], cx
                                <1>       ;jmp   sysret
0000E26C EBD6                   <1>       jmp    short sysvideo_1_2
                                <1>
                                <1> sysvideo_2_2:
                                <1>       ; bl = 3 (exchange/swap) complete display page
                                <1>       ; esi = video page start address
                                <1>       ; edx = user's buffer address
                                <1>
                                <1>       ;mov   ecx, 4096
0000E26E 89F7                   <1>       mov    edi, esi ; video page start address
0000E270 89D6                   <1>       mov    esi, edx ; user's (new page) buffer address
0000E272 EB04                   <1>       jmp    short sysvideo_2_3
                                <1> sysvideo_3:
                                <1>       ; bl = 1 (or bl = 3, stage 2)
                                <1>       ; user to system video/display page transfer (mode 0)
0000E274 89CE                   <1>       mov    esi, ecx ; user buffer
                                <1>       ; edi = video page address
                                <1>       ; 21/11/2020
0000E276 89C1                   <1>       mov    ecx, eax ; 4096
                                <1> sysvideo_2_3:
0000E278 E8F1370000             <1>       call   transfer_from_user_buffer ; fast transfer
                                <1>       ;jc    sysret ; [u.r0] = 0
0000E27D 72C5                   <1>       jc     short sysvideo_1_2 ; 28/01/2021
0000E27F 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.
0000E281 80E303                 <1>       and    bl, 3
0000E284 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)
0000E28A 81FE00800B00           <1>       cmp    esi, 0B8000h ; source buffer address (system)
0000E290 0F8236F5FFFF           <1>       jb     sysret
0000E296 81FE00000C00           <1>       cmp    esi, 0B8000h+(4096*8)
0000E29C 0F832AF5FFFF           <1>       jnb    sysret
0000E2A2 81FF00800B00           <1>       cmp    edi, 0B8000h ; destination buffer address (system)
0000E2A8 0F821EF5FFFF           <1>       jb     sysret
```

```
6707 0000E2AE 81FF00000C00        <1>       cmp   edi, 0B8000h+(4096*8)
6708 0000E2B4 0F8312F5FFFF        <1>       jnb   sysret
6709                              <1>       ;
6710 0000E2BA 51                  <1>       push  ecx ; X1 and Y1 position - top left column, row
6711 0000E2BB 0FB7C1              <1>       movzx eax, cx ; top left column
6712                              <1>       ; 21/11/2020
6713 0000E2BE C1E910              <1>       shr   ecx, 16 ; top row
6714 0000E2C1 740E                <1>       jz    short sysvideo_4_0 ; bypass following code
6715 0000E2C3 52                  <1>       push  edx ; X2 and Y2 position - bottom right column, row
6716 0000E2C4 50                  <1>       push  eax
6717 0000E2C5 66B8A000            <1>       mov   ax, 80*2 ; 80 colums, 160 bytes per row
6718 0000E2C9 F7E1                <1>       mul   ecx
6719                              <1>             ; eax = offset for start row number
6720 0000E2CB 01C6                <1>       add   esi, eax
6721 0000E2CD 01C7                <1>       add   edi, eax
6722 0000E2CF 58                  <1>       pop   eax
6723 0000E2D0 5A                  <1>       pop   edx
6724                              <1> sysvideo_4_0:
6725 0000E2D1 66D1E0              <1>       shl   ax, 1 ; * 2 ; convert start column number to offset
6726 0000E2D4 7404                <1>       jz    short sysvideo_4_1
6727 0000E2D6 01C6                <1>       add   esi, eax
6728 0000E2D8 01C7                <1>       add   edi, eax
6729                              <1>             ; esi = source page window start offset
6730                              <1>             ; edi = destination page window start offset
6731                              <1> sysvideo_4_1:
6732 0000E2DA 59                  <1>       pop   ecx
6733                              <1>       ;mov  eax, 0B8000h+(80*25*2*8)
6734                              <1>       ; 21/11/2020
6735 0000E2DB B800000C00          <1>       mov   eax, 0B8000h+(4096*8)
6736 0000E2E0 39C6                <1>       cmp   esi, eax
6737 0000E2E2 0F83E4F4FFFF        <1>       jnb   sysret ; out of video page
6738 0000E2E8 39C6                <1>       cmp   esi, eax
6739 0000E2EA 0F83DCF4FFFF        <1>       jnb   sysret  ;out of video page
6740                              <1>
6741 0000E2F0 56                  <1>       push  esi ; ****
6742 0000E2F1 57                  <1>       push  edi ; ***
6743 0000E2F2 52                  <1>       push  edx ; **
6744 0000E2F3 51                  <1>       push  ecx ; *
6745 0000E2F4 C1E910              <1>       shr   ecx, 16 ; top row
6746 0000E2F7 C1EA10              <1>       shr   edx, 16 ; bottom row
6747                              <1>       ; 21/11/2020
6748                              <1>       ;cmp  ecx, 24 ; max. 25 rows
6749 0000E2FA 6683F918            <1>       cmp   cx, 24
6750 0000E2FE 7778                <1>       ja    short sysvideo_6 ; invalid, [u.r0] = 0
6751                              <1>       ;cmp  edx, 24 ; max. 25 rows
6752 0000E300 6683FA18            <1>       cmp   dx, 24
6753 0000E304 7772                <1>       ja    short sysvideo_6 ; invalid, [u.r0] = 0
6754 0000E306 28CA                <1>       sub   dl, cl ; end >= start
6755 0000E308 726E                <1>       jc    short sysvideo_6 ; invalid, [u.r0] = 0
6756                              <1>       ; 21/11/2020
6757 0000E30A 89D3                <1>       mov   ebx, edx ; row count - 1
6758 0000E30C 7415                <1>       jz    short sysvideo_4_2
6759 0000E30E 50                  <1>       push  eax ; *****
6760 0000E30F B8A0000000          <1>       mov   eax, 80*2 ; bytes per row
6761 0000E314 F7E3                <1>       mul   ebx ; 21/11/2020
6762                              <1>             ; eax = window end offset
6763                              <1>             ; (for the last row, before adding column bytes)
6764 0000E316 01C6                <1>       add   esi, eax
6765 0000E318 01C7                <1>       add   edi, eax
6766 0000E31A 58                  <1>       pop   eax ; ***** ; mode 3 video memory end (0C000h)
6767 0000E31B 39C6                <1>       cmp   esi, eax
6768                              <1>       ;ja   short sysvideo_6 ; invalid, [u.r0] = 0
6769 0000E31D 7359                <1>       jnb   short sysvideo_6 ; 21/11/2020
6770 0000E31F 39C7                <1>       cmp   edi, eax
6771                              <1>       ;ja   short sysvideo_6 ; invalid, [u.r0] = 0
6772 0000E321 7355                <1>       jnb   short sysvideo_6 ; 21/11/2020
6773                              <1> sysvideo_4_2:
6774 0000E323 59                  <1>       pop   ecx ; *
6775 0000E324 5A                  <1>       pop   edx ; **
6776 0000E325 81E1FFFF0000        <1>       and   ecx, 0FFFFh
6777 0000E32B 81E2FFFF0000        <1>       and   edx, 0FFFFh
6778                              <1>       ; 21/11/2020
6779                              <1>       ;cmp  ecx, 79 ; max. 80 columns
6780 0000E331 6683F94F            <1>       cmp   cx, 79
6781 0000E335 7743                <1>       ja    short sysvideo_7 ; invalid, [u.r0] = 0
6782                              <1>       ;cmp  edx, 79 ; max. 80 columns
6783 0000E337 6683FA4F            <1>       cmp   dx, 79
6784 0000E33B 773D                <1>       ja    short sysvideo_7 ; invalid, [u.r0] = 0
6785 0000E33D 28CA                <1>       sub   dl, cl
6786 0000E33F 7639                <1>       jna   short sysvideo_7 ; invalid, [u.r0] = 0
6787                              <1>       ; 21/11/2020
6788 0000E341 740E                <1>       jz    short sysvideo_4_3
6789                              <1>       ; edx = column count (width) - 1
6790 0000E343 D0E2                <1>       shl   dl, 1 ; * 2 ; byte offset (in end row)
6791 0000E345 01D6                <1>       add   esi, edx
6792 0000E347 01D7                <1>       add   edi, edx
6793                              <1>             ; esi = source page window end offset
6794                              <1>             ; edi = destination page window end offset
6795 0000E349 39C6                <1>       cmp   esi, eax ; video memory end
6796                              <1>       ;ja   short sysvideo_7
6797 0000E34B 732D                <1>       jnb   short sysvideo_7 ; 21/11/2020
6798 0000E34D 39C7                <1>       cmp   edi, eax ; video memory end
6799                              <1>       ;ja   short sysvideo_7
6800 0000E34F 7329                <1>       jnb   short sysvideo_7 ; 21/11/2020
6801                              <1> sysvideo_4_3:
6802 0000E351 5F                  <1>       pop   edi ; ***
6803 0000E352 5E                  <1>       pop   esi ; ****
6804 0000E353 FEC3                <1>       inc   bl ; row count - 1 -> row count
6805 0000E355 FEC2                <1>       inc   dl ; column count
6806 0000E357 88D7                <1>       mov   bh, dl
6807 0000E359 D0E2                <1>       shl   dl, 1 ; convert column count to byte offset
6808                              <1>       ; 21/11/2020
6809                              <1>       ; esi = source page window start offset
6810                              <1>       ; edi = destination page window start offset
6811 0000E35B B8A0000000          <1>       mov   eax, 80*2 ; bytes per row
```

```
6812                                <1>       ; Note: 160 bytes per row (even if move count < 160)
6813                                <1> sysvideo_5:
6814 0000E360 88F9                 <1>       mov    cl, bh ; move/transfer -word- count per row
6815 0000E362 0115[64030300]       <1>       add    [u.r0], edx ; transfer count in bytes
6816 0000E368 F366A5               <1>       rep    movsw
6817 0000E36B 01C6                 <1>       add    esi, eax ; + 160 bytes to next row
6818 0000E36D 01C7                 <1>       add    edi, eax ; + 160 bytes to next row
6819 0000E36F FECB                 <1>       dec    bl ; remain count of rows
6820 0000E371 75ED                 <1>       jnz    short sysvideo_5
6821 0000E373 E954F4FFFF           <1>       jmp    sysret
6822                                <1>
6823                                <1> sysvideo_6:
6824 0000E378 59                   <1>       pop    ecx ; *
6825 0000E379 5A                   <1>       pop    edx ; **
6826                                <1> sysvideo_7:
6827 0000E37A 5F                   <1>       pop    edi ; ***
6828 0000E37B 5E                   <1>       pop    esi ; ****
6829                                <1> sysvideo_8:
6830 0000E37C E94BF4FFFF           <1>       jmp    sysret
6831                                <1>
6832                                <1> sysvideo_9:
6833                                <1>       ; user to system or system to user window transfer
6834                                <1>       ; 28/01/2021 (bl = 3 -> swap/exchange)
6835                                <1>       ;cmp   bl, 2
6836                                <1>       ;;ja   sysret ; user to user transfer is invalid
6837                                <1>       ;             ; [u.r0] = 0
6838                                <1>       ;ja    short sysvideo_8 ; 26/12/2020
6839                                <1>
6840                                <1>       ; 28/01/2021
6841 0000E381 80FB02               <1>       cmp    bl, 2
6842 0000E384 7604                 <1>       jna    short sysvideo_9_8
6843                                <1>
6844                                <1>       ; swap/ exchange video memory and user mem windows
6845                                <1>       ; edi = swap address in user's memory space
6846 0000E386 21FF                 <1>       and    edi, edi
6847 0000E388 74F2                 <1>       jz     short sysvideo_8 ; invalid ; 28/01/2021
6848                                <1>
6849                                <1> sysvideo_9_8:
6850 0000E38A 56                   <1>       push   esi ; ****
6851 0000E38B 57                   <1>       push   edi ; ***
6852 0000E38C 52                   <1>       push   edx ; **
6853 0000E38D 51                   <1>       push   ecx ; *
6854                                <1>
6855 0000E38E C1E910               <1>       shr    ecx, 16 ; top row
6856 0000E391 C1EA10               <1>       shr    edx, 16 ; bottom row
6857                                <1>
6858                                <1>       ; 21/11/2020
6859                                <1>       ;cmp   ecx, 24 ; max. 25 rows
6860 0000E394 6683F918             <1>       cmp    cx, 24
6861 0000E398 77DE                 <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
6862                                <1>       ;cmp   edx, 24 ; max. 25 rows
6863 0000E39A 6683FA18             <1>       cmp    dx, 24
6864 0000E39E 77D8                 <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
6865 0000E3A0 28CA                 <1>       sub    dl, cl
6866 0000E3A2 72D4                 <1>       jc     short sysvideo_6 ; invalid, [u.r0] = 0
6867                                <1>
6868                                <1>       ;mov   ch, cl ; top row
6869                                <1>       ;mov   cl, [ACTIVE_PAGE]
6870                                <1>
6871                                <1>       ;mov   edi, 80*25*2 ; 4000
6872                                <1>       ; 21/11/2020
6873                                <1>       ;mov   edi, 4096 ; [CRT_LEN = 4096 for video mode 3
6874                                <1>       ;shl   edi, cl  ; ! wrong for page 2 to page 7 !
6875                                <1>       ;;add  edi, 0B8000h - 80*25*2
6876                                <1>       ;add   edi, 0B8000h - 1000h ; - 4096
6877                                <1>
6878                                <1>       ; 21/11/2020
6879                                <1>       ;xor   eax, eax
6880                                <1>       ;mov   edi, 0B8000h
6881                                <1>       ;and   cl, cl ; is video page = 0 ?
6882                                <1>       ;jz    short sysvideo_9_1 ; yes
6883                                <1>       ; eax = 0
6884                                <1>
6885                                <1> ;sysvideo_9_0:
6886                                <1>       ;add   ax, 4096
6887                                <1>       ;dec   cl
6888                                <1>       ;jnz   short sysvideo_9_0
6889                                <1>       ;add   edi, eax
6890                                <1>       ;      ; edi = video page start address
6891                                <1>
6892                                <1>       ; 21/11/2020
6893 0000E3A4 BF00800B00           <1>       mov    edi, 0B8000h
6894 0000E3A9 803D[1E890100]00     <1>       cmp    byte [ACTIVE_PAGE], 0
6895 0000E3B0 760C                 <1>       jna    short sysvideo_9_1
6896                                <1> stsvideo_9_0:
6897 0000E3B2 B010                 <1>       mov    al, 16 ; 4096/256
6898 0000E3B4 F625[1E890100]       <1>       mul    byte [ACTIVE_PAGE]
6899 0000E3BA 86E0                 <1>       xchg   ah, al ; * 256
6900 0000E3BC 01C7                 <1>       add    edi, eax
6901                                <1>              ; edi = video page start address
6902                                <1> sysvideo_9_1:
6903                                <1>       ; bl = transfer direction
6904                                <1>       ;     (1 = from user, 2 = to user)
6905                                <1>       ;     (3 = swap) ; 28/01/2021
6906 0000E3BE 88D7                 <1>       mov    bh, dl ; row count - 1
6907                                <1>       ;mov   dl, ch ; top row
6908                                <1>       ; 21/11/2020
6909 0000E3C0 08C9                 <1>       or     cl, cl ; top row number
6910 0000E3C2 7408                 <1>       jz     short sysvideo_9_2
6911                                <1>
6912                                <1>       ;mov   eax, 80*2
6913 0000E3C4 66B8A000             <1>       mov    ax, 80*2 ; 160, bytes per row
6914 0000E3C8 F7E1                 <1>       mul    ecx ; 22/11/2020
6915 0000E3CA 01C7                 <1>       add    edi, eax
6916                                <1>              ; edi = window start address for top row
```

```
                        <1> sysvideo_9_2:
0000E3CC 59             <1>         pop     ecx ; *
0000E3CD 5A             <1>         pop     edx ; **
0000E3CE 81E1FFFF0000   <1>         and     ecx, 0FFFFh
0000E3D4 81E2FFFF0000   <1>         and     edx, 0FFFFh
                        <1>         ; 21/11/2020
                        <1>         ;cmp    ecx, 79 ; max. 80 columns
0000E3DA 6683F94F       <1>         cmp     cx, 79
0000E3DE 779A           <1>         ja      short sysvideo_7 ; invalid, [u.r0] = 0
                        <1>         ;cmp    edx, 79 ; max. 80 columns
0000E3E0 6683FA4F       <1>         cmp     dx, 79
0000E3E4 7794           <1>         ja      short sysvideo_7 ; invalid, [u.r0] = 0
                        <1>
0000E3E6 28CA           <1>         sub     dl, cl
0000E3E8 7290           <1>         jc      short sysvideo_7 ; invalid, [u.r0] = 0
                        <1>
0000E3EA 08C9           <1>         or      cl, cl ; left column
0000E3EC 7404           <1>         jz      short sysvideo_9_3 ; 0
                        <1>
                        <1>         ; 21/11/2020
0000E3EE D0E1           <1>         shl     cl, 1  ; column * 2
0000E3F0 01CF           <1>         add     edi, ecx
                        <1>                 ; edi = window start addr for top left column
                        <1> sysvideo_9_3:
0000E3F2 88D1           <1>         mov     cl, dl
0000E3F4 FEC1           <1>         inc     cl ; column count
0000E3F6 D0E1           <1>         shl     cl, 1 ; column count * 2
                        <1>                 ; ecx = transfer count per row
                        <1>
0000E3F8 58             <1>         pop     eax ; *** (swap address)
0000E3F9 5E             <1>         pop     esi ; ****
                        <1>
0000E3FA FEC7           <1>         inc     bh  ; row count
                        <1>
                        <1>         ;mov    edx, 80*2
0000E3FC 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
0000E3FE F6C301         <1>         test    bl, 1
0000E401 7439           <1>         jz      short sysvideo_11 ; system to user transfer
                        <1>
                        <1>         ; user to system video/display page window transfer (mode 0)
0000E403 21C0           <1>         and     eax, eax ; swap address
0000E405 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)
0000E407 56             <1>         push    esi ; user buffer
0000E408 57             <1>         push    edi ; beginning address of the window
                        <1>         ; 21/11/2020
0000E409 53             <1>         push    ebx ; save bh
0000E40A 89FE           <1>         mov     esi, edi
0000E40C 89C7           <1>         mov     edi, eax
                        <1> sysvideo_9_4:
0000E40E E811360000     <1>         call    transfer_to_user_buffer ; fast transfer
0000E413 7208           <1>         jc      short sysvideo_9_5
                        <1>         ; ecx = actual transfer count (must be same with input)
0000E415 01D6           <1>         add     esi, edx ; next row address of (video page) window
0000E417 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
0000E419 FECF           <1>         dec     bh
0000E41B 75F1           <1>         jnz     short sysvideo_9_4 ; repeat for next row
                        <1> sysvideo_9_5:
0000E41D 5B             <1>         pop     ebx ; restore bh
0000E41E 5F             <1>         pop     edi
0000E41F 5E             <1>         pop     esi
                        <1>         ;jnc    short sysvideo_10
0000E420 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
0000E422 E847360000     <1>         call    transfer_from_user_buffer ; fast transfer
                        <1>         ;jc     sysret
0000E427 720E           <1>         jc      short sysvideo_9_6 ; 28/01/2021
                        <1>         ; ecx = actual transfer count (must be same with input)
0000E429 010D[64030300] <1>         add     [u.r0], ecx ; actual transfer count
0000E42F 01D7           <1>         add     edi, edx ; next row address of (video page) window
0000E431 01CE           <1>         add     esi, 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
0000E433 FECF           <1>         dec     bh
0000E435 75EB           <1>         jnz     short sysvideo_10 ; repeat for next row
                        <1>         ;jmp    sysret
                        <1> sysvideo_9_6:
0000E437 E990F3FFFF     <1>         jmp     sysret
                        <1>
                        <1> sysvideo_11:
                        <1>         ; system to user video/display page window transfer (mode 0)
0000E43C 87FE           <1>         xchg    edi, esi
                        <1> sysvideo_12:
                        <1>         ; esi = beginning addr of the (screen, video page) window
                        <1>         ; edi =       user's buffer
0000E43E E8E1350000     <1>         call    transfer_to_user_buffer ; fast transfer
0000E443 0F8283F3FFFF   <1>         jc      sysret
                        <1>         ; ecx = actual transfer count (must be same with input)
0000E449 010D[64030300] <1>         add     [u.r0], ecx
0000E44F 01D6           <1>         add     esi, edx ; next row (edx = 160)
0000E451 01CF           <1>         add     edi, ecx ; next row of the user's window
                        <1>                          ; (ecx <= 160)
0000E453 FECF           <1>         dec     bh
```

```
7022 0000E455 75E7                <1>       jnz    short sysvideo_12
7023                              <1> sysvideo_12_0:
7024 0000E457 E970F3FFFF          <1>       jmp    sysret
7025                              <1>
7026                              <1> sysvideo_13:
7027                              <1>       ; 28/12/2020
7028 0000E45C 80FF01              <1>       cmp    bh, 1
7029 0000E45F 7753                <1>         ja      short sysvideo_15 ; 23/11/2020
7030                              <1>
7031                              <1>       ; 25/02/2021
7032                              <1>       ; 12/02/2021
7033                              <1>       ; 29/01/2021, 31/01/2021
7034                              <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
7035                              <1>       ; (major modification, from mode 13h to all VGA modes,
7036                              <1>       ;  except super VGA modes and liner frame buffer method)
7037                              <1>
7038                              <1>       ; BH = 1 = VGA Graphics mode (0A0000h) data transfers
7039                              <1>
7040                              <1>       ; 29/01/2021
7041 0000E461 66B84001            <1>       mov    ax, 320      ; 320 pixels
7042 0000E465 F6C380              <1>       test   bl, 80h      ; bit 7 (screen width, 640 pixels)
7043 0000E468 7408                <1>       jz     short sysvideo_13_0
7044 0000E46A 66D1E0              <1>       shl    ax, 1  ; 640 pixels
7045                              <1>       ;
7046 0000E46D 80E37F              <1>       and    bl, 7Fh
7047 0000E470 7405                <1>       jz     short sysvideo_14
7048                              <1> sysvideo_13_0:
7049                              <1>       ; 29/01/2021
7050 0000E472 80FB41              <1>       cmp    bl, 41h
7051 0000E475 77E0                <1>       ja     short sysvideo_12_0 ; invalid (unknown) sub function
7052                              <1> sysvideo_14:
7053 0000E477 66A3[86120300]      <1>       mov    [v_width], ax; save screen width
7054 0000E47D C705[8A120300]0000- <1>       mov    dword [v_mem], 0A0000h  ; save video memory address
7054 0000E485 0A00                <1>
7055 0000E487 C705[8E120300]0000- <1>       mov    dword [v_siz], 65536      ; save video memory size
7055 0000E48F 0100                <1>
7056 0000E491 C705[96120300]0000- <1>       mov    dword [v_end], 0B0000h    ; save end of video page
7056 0000E499 0B00                <1>
7057 0000E49B B708                <1>       mov    bh, 8
7058 0000E49D 883D[89120300]      <1>       mov    [v_bpp], bh ; 8     ; bits per pixel (256 colors)
7059 0000E4A3 881D[88120300]      <1>       mov    [v_ops], bl  ; VGA data transfer options
7060                              <1>       ;mov   [maskbuff], edi ; 25/02/2021
7061 0000E4A9 893D[9A120300]      <1>       mov    [maskcolor], edi ; 25/02/2021
7062                              <1>                     ; save mask color or bitmask buffer address
7063 0000E4AF E9BD000000          <1>       jmp    sysvideo_15_7
7064                              <1>
7065                              <1> sysvideo_15:
7066                              <1>       ; 28/12/2020
7067 0000E4B4 80FF02              <1>       cmp    bh, 2
7068 0000E4B7 0F87F01E0000        <1>         ja      sysvideo_16
7069                              <1>
7070                              <1>       ; 25/02/2021
7071                              <1>       ; 12/02/2021
7072                              <1>       ; 30/01/2021, 31/01/2021
7073                              <1>       ; 01/01/2021, 29/01/2021
7074                              <1>       ; 26/12/2020, 27/12/2020
7075                              <1>       ; 25/12/2020 (TRDOS 386 v2.0.3)
7076                              <1>       ;
7077                              <1>       ; BH = 2 = SVGA (VESA VBE) Graphics mode (LFB) data transfers
7078                              <1>
7079                              <1>       ; 25/12/2020
7080                              <1>       ; resolution table entry will be saved into EBP register
7081                              <1>
7082 0000E4BD 803D[52090000]02    <1>       cmp    byte [vbe3], 2 ; VESA VBE 3 video bios
7083                              <1>                       ; or BOCHS/QEMU/VIRTUALBOX emu video bios
7084 0000E4C4 724E                <1>       jb     short sysvideo_15_4 ; no, nothing to do !
7085 0000E4C6 770B                <1>       ja     short sysvideo_15_0 ; yes
7086                              <1>
7087                              <1>       ; Only Bochs/Plex86 (emu) vbe2 video bios is usable in pmid
7088                              <1>       ; (if [vbe3] = 2)
7089 0000E4C8 A0[53090000]        <1>       mov    al, [vbe2bios] ; Bochs vbios sign is from C0h to C5h
7090 0000E4CD 24F0                <1>       and    al, 0F0h
7091 0000E4CF 3CC0                <1>       cmp    al, 0C0h
7092 0000E4D1 7541                <1>       jne    short sysvideo_15_4 ; unknown (vbe2) video bios
7093                              <1> sysvideo_15_0:
7094                              <1>       ; 29/01/2021
7095 0000E4D3 80FB41              <1>       cmp    bl, 41h
7096 0000E4D6 773C                <1>       ja     short sysvideo_15_4 ; invalid (unknown) sub function
7097                              <1>       ; 29/01/2021
7098 0000E4D8 881D[88120300]      <1>       mov    [v_ops], bl  ; SVGA data transfer options
7099                              <1>
7100 0000E4DE 89D8                <1>       mov    eax, ebx ; hw of ebx is vesa vbe video mode
7101 0000E4E0 C1E810              <1>       shr    eax, 16 ; ax = vesa vbe video mode
7102 0000E4E3 7513                <1>       jnz    short sysvideo_15_2
7103                              <1>       ; ax = 0
7104                              <1>
7105                              <1>       ; check & use current video mode
7106 0000E4E5 803D[9A6E0000]FF    <1>       cmp    byte [CRT_MODE], 0FFh ; extended (SVGA) mode ?
7107 0000E4EC 7526                <1>       jne    short sysvideo_15_4 ; no
7108                              <1> sysvideo_15_1:
7109                              <1>       ; use current vbe (svga) video mode
7110 0000E4EE 66A1[1E120300]      <1>       mov    ax, [video_mode] ; extended (SVGA, VESA VBE) mode
7111 0000E4F4 6625FF01            <1>       and    ax, 1FFh ; vesa vbe video mode: 1XXh
7112                              <1> sysvideo_15_2:
7113                              <1>       ; 29/01/2021
7114                              <1>       ;mov   [maskbuff], edi ; 25/02/2021
7115 0000E4F8 893D[9A120300]      <1>       mov    [maskcolor], edi ; 25/02/2021
7116                              <1>                     ; save mask color or bitmask buffer address
7117 0000E4FE BD[26720000]        <1>       mov    ebp, b_vbe_modes ; vbe mode table (in 'vidata.s')
7118                              <1> sysvideo_15_3:
7119 0000E503 663B4500            <1>       cmp    ax, [ebp]
7120 0000E507 7410                <1>       je     short sysvideo_15_5
7121 0000E509 83C508              <1>       add    ebp, 8 ; vbe mode table entry size
7122 0000E50C 81FD[E6720000]      <1>       cmp    ebp, end_of_b_vbe_modes
7123 0000E512 72EF                <1>       jb     short sysvideo_15_3
```

```
                            <1> sysvideo_15_4:
                            <1>       ; desired video mode is not a valid (implemented)
                            <1>       ;       extended (VESA VBE, SVGA) video mode
                            <1>       ;
                            <1>       ; nothing to do !
                            <1>
                            <1>       ; [u.r0] = 0  ; return value of EAX
0000E514 E9B3F2FFFF         <1>       jmp   sysret
                            <1>
                            <1> sysvideo_15_5:
                            <1>       ; get LFB address
0000E519 A1[2C120300]       <1>       mov   eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
0000E51E 09C0               <1>       or    eax, eax
0000E520 7509               <1>       jnz   short sysvideo_15_6
0000E522 66A1[E10E0000]     <1>       mov   ax, [def_LFB_addr] ; default LFB addr
                            <1>                                ; (for vbe mode 118h)
0000E528 C1E010             <1>       shl   eax, 16
                            <1>       ; 27/12/2020
                            <1>       ;jz   short sysvideo_15_4
                            <1> sysvideo_15_6:
                            <1>       ; 29/01/2021
0000E52B A3[8A120300]       <1>       mov   [v_mem], eax ; save video memory address
                            <1>
                            <1>       ; 27/12/2020
                            <1>       ; 26/12/2020
0000E530 8B4502             <1>       mov   eax, [ebp+2] ; width, height
                            <1>       ; 29/01/2021
0000E533 66A3[86120300]     <1>       mov   [v_width], ax ; save screen width
                            <1>       ; 28/12/2020
0000E539 8A7D06             <1>       mov   bh, [ebp+6] ; bpp
                            <1>       ; 28/02/2021
                            <1>       ; check default truecolor bpp value and use
                            <1>       ; 32bpp instead of 24bpp if the default value
                            <1>       ; has been set to 32bpp.
0000E53C 80FF18             <1>       cmp   bh, 24
0000E53F 750B               <1>       jne   short sysvideo_15_16
0000E541 803D[B7850100]20   <1>       cmp   byte [truecolor], 32
                            <1>             ; Default truecolor bpp value,
                            <1>             ; it is 32 for VBE3 video bios
                            <1>             ; (it can be set to 32 or 24)
0000E548 7502               <1>       jne   short sysvideo_15_16 ; not VBE3 !
                            <1>                                  ; or it is set to 24
0000E54A B720               <1>       mov   bh, 32
                            <1>       ; 28/02/2021
                            <1> sysvideo_15_16:
                            <1>       ; 29/01/2021
0000E54C 883D[89120300]     <1>       mov   [v_bpp], bh ; bits per pixel
                            <1>
0000E552 52                 <1>       push  edx ; *
0000E553 0FB7D0             <1>       movzx edx, ax  ; width
0000E556 C1E810             <1>       shr   eax, 16  ; height
0000E559 F7E2               <1>       mul   edx
                            <1>       ; eax = linear frame buffer size (pixels)
                            <1>       ; 29/01/2021
0000E55B A3[8E120300]       <1>       mov   [v_siz], eax ; save video page size
0000E560 E8FD000000         <1>       call  pixels_to_byte_count
0000E565 0305[8A120300]     <1>       add   eax, [v_mem]
0000E56B A3[96120300]       <1>       mov   [v_end], eax ; save end of video page
0000E570 5A                 <1>       pop   edx ; *
                            <1>
                            <1>       ; bh = bits per pixel
                            <1>       ; (bh will not be used after here, 29/01/2021)
                            <1>
                            <1>       ; bl = pixel operations & options
                            <1>       ; ecx, edx, esi, edi input parameters
                            <1>       ; [maskcolor] = edi input ; 25/02/2021
                            <1>
                            <1> sysvideo_15_7:
                            <1>       ; 29/01/2021
                            <1>       ;test  byte [v_ops], 40h  ; system to user ?
0000E571 F6C340             <1>       test  bl, 40h
0000E574 7517               <1>       jnz   short sysvideo_15_9
                            <1>
0000E576 31C0               <1>       xor   eax, eax
0000E578 88D8               <1>       mov   al, bl
0000E57A BB[A0E60000]       <1>       mov   ebx, pixel_ops
0000E57F 240F               <1>       and   al, 0Fh ; isolate 16 pixel operations
0000E581 C0E002             <1>       shl   al, 2 ; * 4 for dword table pointers
0000E584 01C3               <1>       add   ebx, eax
                            <1>
                            <1>       ; ebx = subroutine address
                            <1>
                            <1>       ; ecx, edx, esi, edi input parameters
                            <1>       ; [maskbuff] = edi input
                            <1>       ; [maskcolor] = edi input ; 25/02/2021
                            <1>
0000E586 FF13               <1>       call  [ebx]
                            <1> sysvideo_15_8:
0000E588 E93FF2FFFF         <1>       jmp   sysret
                            <1>
                            <1> sysvideo_15_9:
                            <1>       ; system to user display page or window copy
                            <1>       ;test  byte [v_ops], 1 ; window copy ?
0000E58D F6C301             <1>       test  bl, 1
0000E590 7521               <1>       jnz   short sysvideo_15_10
                            <1>
                            <1>       ; display page (full screen copy)
0000E592 8B35[8A120300]     <1>       mov   esi, [v_mem] ; LFB start address
0000E598 A1[8E120300]       <1>       mov   eax, [v_siz]
0000E59D E8C0000000         <1>       call  pixels_to_byte_count
0000E5A2 89C1               <1>       mov   ecx, eax ; transfer count in bytes
                            <1>       ;edi = user's buffer address
0000E5A4 E87B340000         <1>       call  transfer_to_user_buffer
0000E5A9 72DD               <1>       jc    short sysvideo_15_8
0000E5AB 890D[64030300]     <1>       mov   [u.r0], ecx
```

```
7229 0000E5B1 EBD5                <1>       jmp    short sysvideo_15_8
7230                              <1>
7231                              <1> sysvideo_15_10:
7232 0000E5B3 E820000000          <1>       call   sysvideo_15_12 ; window preparations
7233 0000E5B8 72CE                <1>       jc     short sysvideo_15_8
7234                              <1>
7235 0000E5BA 8B35[92120300]      <1>       mov    esi, [v_str]
7236                              <1> sysvideo_15_11:
7237                              <1>       ; esi = window's current row address (video mem)
7238                              <1>       ; edi = current row (virtual) addr in user's buff
7239                              <1>       ; ecx = transfer count per row
7240 0000E5C0 E85F340000          <1>       call   transfer_to_user_buffer
7241 0000E5C5 72C1                <1>       jc     short sysvideo_15_8
7242 0000E5C7 010D[64030300]      <1>       add    [u.r0], ecx
7243 0000E5CD 4B                  <1>       dec    ebx
7244 0000E5CE 74B8                <1>       jz     short sysvideo_15_8 ; ok.
7245                              <1>       ; next row
7246 0000E5D0 01CF                <1>       add    edi, ecx ; next row in user's buffer
7247 0000E5D2 01D6                <1>       add    esi, edx ; next row of window (system)
7248 0000E5D4 EBEA                <1>       jmp    short sysvideo_15_11
7249                              <1>
7250                              <1> sysvideo_15_14:
7251 0000E5D6 F9                  <1>       stc    ; error !
7252                              <1> sysvideo_15_15:
7253 0000E5D7 C3                  <1>       retn
7254                              <1>
7255                              <1> sysvideo_15_12:
7256                              <1>       ; 30/01/2021
7257                              <1>       ; 29/01/2021
7258                              <1>       ; Window address preparations for window copy
7259 0000E5D8 6621D2              <1>       and    dx, dx
7260 0000E5DB 74F9                <1>       jz     short sysvideo_15_14 ; invalid (zero columns)
7261                              <1>       ;test  edx, 0FFFF0000h
7262                              <1>       ;jz    short sysvideo_15_14 ; invalid (zero rows)
7263 0000E5DD 81FA00000100        <1>       cmp    edx, 65536
7264 0000E5E3 72F2                <1>       jb     short sysvideo_15_15 ; invalid (zero rows)
7265 0000E5E5 89C8                <1>       mov    eax, ecx ; start position (row, column)
7266 0000E5E7 E899000000          <1>       call   calc_pixel_offset
7267 0000E5EC 3B05[8E120300]      <1>       cmp    eax, [v_siz]
7268 0000E5F2 73E2                <1>       jnb    short sysvideo_15_14 ; out of display page
7269                              <1>                           ; nothing to do
7270 0000E5F4 E869000000          <1>       call   pixels_to_byte_count
7271 0000E5F9 0305[8A120300]      <1>       add    eax, [v_mem]
7272 0000E5FF A3[92120300]        <1>       mov    [v_str], eax ; window start address
7273                              <1>                           ; (addr of top left corner)
7274                              <1>       ; check column limit
7275 0000E604 89C8                <1>       mov    eax, ecx
7276 0000E606 6601D0              <1>       add    ax, dx  ; add columns to start column
7277 0000E609 72CC                <1>       jc     short sysvideo_15_15 ; cf = 1
7278 0000E60B 663B05[86120300]    <1>       cmp    ax, [v_width]
7279 0000E612 77C2                <1>       ja     short sysvideo_15_14
7280                              <1>
7281 0000E614 89D0                <1>       mov    eax, edx ; size
7282 0000E616 2D00000100          <1>       sub    eax, 65536 ; row count -> 0 based row #
7283 0000E61B E865000000          <1>       call   calc_pixel_offset
7284 0000E620 3B05[8E120300]      <1>       cmp    eax, [v_siz] ; video (display) page size
7285 0000E626 77AE                <1>       ja     short sysvideo_15_14 ; out of display page
7286                              <1>                           ; nothing to do
7287 0000E628 E835000000          <1>       call   pixels_to_byte_count
7288 0000E62D 0305[92120300]      <1>       add    eax, [v_str] ; window start address
7289 0000E633 3B05[96120300]      <1>       cmp    eax, [v_end] ; window end address (+1)
7290                              <1>                     ; (addr of bottom right corner +1)
7291 0000E639 779B                <1>       ja     short sysvideo_15_14 ; out of display page
7292                              <1>                           ; nothing to do
7293 0000E63B 89D3                <1>       mov    ebx, edx
7294 0000E63D C1EB10              <1>       shr    ebx, 16
7295                              <1>       ; ebx = row count
7296 0000E640 81E2FFFF0000        <1>       and    edx, 0FFFFh
7297                              <1>       ; edx = transfer count per row (from user's buffer)
7298                              <1>       ;      (in pixels, window width)
7299 0000E646 89D0                <1>       mov    eax, edx
7300 0000E648 A3[9E120300]        <1>       mov    [pixcount], eax ; 27/02/2021
7301 0000E64D E810000000          <1>       call   pixels_to_byte_count
7302 0000E652 89C1                <1>       mov    ecx, eax
7303                              <1>       ; ecx = transfer count per row (from user's buffer)
7304                              <1>       ;      (in bytes, window width)
7305 0000E654 66A1[86120300]      <1>       mov    ax, [v_width]
7306 0000E65A E803000000          <1>       call   pixels_to_byte_count
7307 0000E65F 89C2                <1>       mov    edx, eax
7308                              <1>       ; edx = byte count per row
7309 0000E661 C3                  <1>       retn ; cf = 0
7310                              <1>
7311                              <1> pixels_to_byte_count:
7312                              <1>       ; 29/01/2021
7313                              <1>       ; INPUT:
7314                              <1>       ;   eax = pixel count
7315                              <1>       ; OUTPUT:
7316                              <1>       ;   eax = byte count
7317                              <1>       ;
7318 0000E662 803D[89120300]08    <1>       cmp    byte [v_bpp], 8
7319 0000E669 7619                <1>       jna    short pixtobc_3 ; 8 bit colors
7320 0000E66B 803D[89120300]18    <1>       cmp    byte [v_bpp], 24
7321 0000E672 720A                <1>       jb     short pixtobc_1 ; 16 bit colors
7322 0000E674 770B                <1>       ja     short pixtobc_2 ; 32 bit colors
7323                              <1>       ; 24 bit pixels
7324                              <1>       ; eax = eax * 3
7325                              <1>       ;push  edx
7326                              <1>       ;mov   edx, eax
7327                              <1>       ;shl   eax, 1
7328                              <1>       ;add   eax, edx
7329                              <1>       ;pop   edx
7330 0000E676 50                  <1>       push   eax
7331 0000E677 D1E0                <1>       shl    eax, 1
7332 0000E679 010424              <1>       add    [esp], eax
7333 0000E67C 58                  <1>       pop    eax
```

```
7334 0000E67D C3                <1>       retn
7335                            <1> pixtobc_1:
7336                            <1>       ; 32 bit pixels
7337                            <1>       ; eax = eax * 2
7338 0000E67E D1E0              <1>       shl    eax, 1
7339 0000E680 C3                <1>       retn
7340                            <1> pixtobc_2:
7341                            <1>       ; 16 bit pixels
7342                            <1>       ; eax = eax * 4
7343 0000E681 C1E002            <1>       shl    eax, 2
7344                            <1> pixtobc_3:
7345 0000E684 C3                <1>       retn
7346                            <1>
7347                            <1> calc_pixel_offset:
7348                            <1>       ; 29/01/2021
7349                            <1>       ; INPUT:
7350                            <1>       ;   eax = pixel position (row, column)
7351                            <1>       ; OUTPUT:
7352                            <1>       ;   eax = pixel offset (linear address)
7353                            <1>       ;
7354 0000E685 52                <1>       push   edx
7355 0000E686 50                <1>       push   eax
7356 0000E687 C1E810            <1>       shr    eax, 16
7357 0000E68A 7409              <1>       jz     short cpixo_0
7358                            <1>       ; eax = row
7359 0000E68C 0FB715[86120300]  <1>       movzx  edx, word [v_width]
7360 0000E693 F7E2              <1>       mul    edx
7361                            <1> cpixo_0:
7362                            <1>       ; eax = row * screen width
7363 0000E695 5A                <1>       pop    edx
7364 0000E696 81E2FFFF0000      <1>       and    edx, 0FFFFh
7365                            <1>       ; edx = column
7366 0000E69C 01D0              <1>       add    eax, edx
7367                            <1>       ; eax = (row * screen width) + column
7368 0000E69E 5A                <1>       pop    edx
7369 0000E69F C3                <1>       retn
7370                            <1>
7371                            <1>       ; 02/02/2021
7372                            <1>       ; 29/01/2021
7373                            <1> pixel_ops:
7374 0000E6A0 [E0E60000]        <1>       dd     pix_op_cpy ; copy pixels (user to system)
7375 0000E6A4 [2BEC0000]        <1>       dd     pix_op_new ; change (new, fill) color
7376 0000E6A8 [48E70000]        <1>       dd     pix_op_add ; add color (up to 0FFh)
7377 0000E6AC [FAE70000]        <1>       dd     pix_op_sub ; sub color (down to 0)
7378 0000E6B0 [15EA0000]        <1>       dd     pix_op_orc ; or color
7379 0000E6B4 [C7EA0000]        <1>       dd     pix_op_and ; and color
7380 0000E6B8 [79EB0000]        <1>       dd     pix_op_xor ; xor color
7381 0000E6BC [EFEC0000]        <1>       dd     pix_op_not ; not color
7382 0000E6C0 [99ED0000]        <1>       dd     pix_op_neg ; neg color
7383 0000E6C4 [43EE0000]        <1>       dd     pix_op_inc ; inc color
7384 0000E6C8 [EDEE0000]        <1>       dd     pix_op_dec ; dec color
7385 0000E6CC [ACE80000]        <1>       dd     pix_op_mix ; mix color
7386 0000E6D0 [73E90000]        <1>       dd     pix_op_rpl ; replace color
7387 0000E6D4 [97EF0000]        <1>       dd     pix_op_blk ; copy pixel block(s) (sys)
7388 0000E6D8 [46F00000]        <1>       dd     pix_op_lin ; write line(s)
7389 0000E6DC [29F40000]        <1>       dd     pix_op_chr ; write character (font)
7390                            <1>
7391                            <1> pix_op_cpy:
7392                            <1>       ; 21/02/2021
7393                            <1>       ; 06/02/2021
7394                            <1>       ; 30/01/2021
7395                            <1>       ; COPY PIXELS
7396                            <1>       ;
7397                            <1>       ; INPUT:
7398                            <1>       ;  If bit 4 of BL or [v_ops] = 1 -window copy-
7399                            <1>       ;  ECX = start position (row, column)
7400                            <1>       ;         (HW = row, CX = column)
7401                            <1>       ;  EDX = size (rows, colums)
7402                            <1>       ;         (HW = rows, DX = columns)
7403                            <1>       ;       (0 -> invalid
7404                            <1>       ;         (1 -> horizontal or vertical line)
7405                            <1>       ;  If bit 4 of BL or [v_ops] = 0 -full screen-
7406                            <1>       ;     ECX and EDX will not be used
7407                            <1>       ;  ESI = user's buffer address
7408                            <1>       ;  [maskcolor] = mask color (to be excluded)
7409                            <1>       ;
7410                            <1>       ; OUTPUT:
7411                            <1>       ;      [u.r0] will be > 0 if succesful
7412                            <1>
7413 0000E6E0 F605[88120300]10  <1>       test   byte [v_ops], 10h ; display page or window ?
7414 0000E6E7 752E              <1>       jnz    short pix_op_cpy_w ; window
7415                            <1>
7416 0000E6E9 8B3D[8A120300]    <1>       mov    edi, [v_mem] ; 21/02/2021
7417                            <1>
7418                            <1>       ; Copy user's buffer content do display page
7419                            <1>       ; (full screen copy)
7420 0000E6EF A1[8E120300]      <1>       mov    eax, [v_siz] ; video page size
7421 0000E6F4 E869FFFFFF        <1>       call   pixels_to_byte_count
7422 0000E6F9 89C1              <1>       mov    ecx, eax ; transfer count
7423                            <1>       ; esi = user's buffer address (virtual)
7424 0000E6FB F605[88120300]20  <1>       test   byte [v_ops], 20h ; masked copy ?
7425 0000E702 7405              <1>       jz     short pix_op_cpy_0 ; no
7426 0000E704 E9720F0000        <1>       jmp    m_pix_op_cpy ; copy pixels except mask color
7427                            <1> pix_op_cpy_0:
7428                            <1>       ; esi = user buffer for full screen copy
7429                            <1>       ; edi = start of video memory
7430                            <1>       ;      (start of display page)
7431                            <1>       ; ecx = byte count (display page size in bytes)
7432 0000E709 E860330000        <1>       call   transfer_from_user_buffer
7433 0000E70E 7206              <1>       jc     short pix_op_cpy_1
7434 0000E710 890D[64030300]    <1>       mov    [u.r0], ecx
7435                            <1> pix_op_cpy_1:
7436 0000E716 C3                <1>       retn   ; 06/02/2021
7437                            <1>
7438                            <1> pix_op_cpy_w:
```

```
7439 0000E717 E8BCFEFFFF          <1>       call   sysvideo_15_12 ; window preparations
7440 0000E71C 72F8                <1>       jc     short pix_op_cpy_1
7441                              <1>       ; ecx = bytes per row (to be applied)
7442                              <1>       ; edx = screen width in bytes
7443                              <1>       ; ebx = row count
7444 0000E71E 8B3D[92120300]      <1>       mov    edi, [v_str]
7445 0000E724 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked copy ?
7446 0000E72B 7405                <1>       jz     short pix_op_cpy_w_0 ; no
7447 0000E72D E90C100000          <1>       jmp    m_pix_op_cpy_w ; window copy except mask color
7448                              <1> pix_op_cpy_w_0:
7449                              <1>       ; esi = current row (virtual) addr in user's buff
7450                              <1>       ; edi = window's current row address (video mem)
7451                              <1>       ; ecx = transfer count per row
7452 0000E732 E837330000          <1>       call   transfer_from_user_buffer
7453 0000E737 72DD                <1>       jc     short pix_op_cpy_1
7454 0000E739 010D[64030300]      <1>       add    [u.r0], ecx
7455 0000E73F 4B                  <1>       dec    ebx
7456 0000E740 74D4                <1>       jz     short pix_op_cpy_1 ; ok.
7457                              <1>       ; next row
7458 0000E742 01CE                <1>       add    esi, ecx ; next row in user's buffer
7459 0000E744 01D7                <1>       add    edi, edx ; next row of window (system)
7460 0000E746 EBEA                <1>       jmp    short pix_op_cpy_w_0
7461                              <1>
7462                              <1> pix_op_add:
7463                              <1>       ; 31/01/2021
7464                              <1>       ; 30/01/2021
7465                              <1>       ; ADD COLOR
7466                              <1>       ;
7467                              <1>       ; INPUT:
7468                              <1>       ;   CL = color (8 bit, 256 colors)
7469                              <1>       ;  ECX = color (16 bit and true colors)
7470                              <1>       ;  EDX = start position (row, column)
7471                              <1>       ;        (HW = row, DX = column)
7472                              <1>       ;  ESI = size (rows, colums)
7473                              <1>       ;        (HW = rows, SI = columns)
7474                              <1>       ;
7475                              <1>       ;  [maskcolor] = mask color (to be excluded)
7476                              <1>       ;
7477                              <1>       ; OUTPUT:
7478                              <1>       ;      [u.r0] will be > 0 if succesful
7479                              <1>
7480 0000E748 F605[88120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
7481 0000E74F 7555                <1>       jnz    short pix_op_add_w ; window
7482                              <1>
7483 0000E751 8B3D[8A120300]      <1>       mov    edi, [v_mem]
7484 0000E757 89FE                <1>       mov    esi, edi
7485                              <1>       ; ecx = color (CL, CX, ECX)
7486 0000E759 89C8                <1>       mov    eax, ecx
7487 0000E75B 8B0D[8E120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
7488                              <1>
7489 0000E761 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color adding ?
7490 0000E768 7405                <1>       jz     short pix_op_add_0 ; no
7491 0000E76A E9CE100000          <1>       jmp    m_pix_op_add ; add color except mask color
7492                              <1> pix_op_add_0:
7493 0000E76F 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7494 0000E776 7707                <1>       ja     short pix_op_add_1
7495                              <1>
7496                              <1>       ; 256 colors (8bpp)
7497 0000E778 E84D0A0000          <1>       call   pix_op_add_8
7498 0000E77D EB1E                <1>       jmp    short pix_op_add_4
7499                              <1>
7500                              <1> pix_op_add_1:
7501 0000E77F 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7502 0000E786 7710                <1>       ja     short pix_op_add_3 ; 32bpp
7503 0000E788 7207                <1>       jb     short pix_op_add_2 ; 16bpp
7504                              <1>
7505                              <1>       ; 24 bit true colors
7506 0000E78A E85B0A0000          <1>       call   pix_op_add_24
7507 0000E78F EB0C                <1>       jmp    short pix_op_add_4
7508                              <1>
7509                              <1>       ; 65536 colors (16bpp)
7510                              <1> pix_op_add_2:
7511 0000E791 E8420A0000          <1>       call   pix_op_add_16
7512 0000E796 EB05                <1>       jmp    short pix_op_add_4
7513                              <1>
7514                              <1>       ; 32 bit true colors
7515                              <1> pix_op_add_3:
7516 0000E798 E86D0A0000          <1>       call   pix_op_add_32
7517                              <1> pix_op_add_4:
7518 0000E79D 29F7                <1>       sub    edi, esi
7519 0000E79F 893D[64030300]      <1>       mov    [u.r0], edi
7520                              <1> pix_op_add_5:
7521 0000E7A5 C3                  <1>       retn
7522                              <1>
7523                              <1> pix_op_add_w:
7524                              <1>       ; 31/01/2021
7525 0000E7A6 51                  <1>       push   ecx ; * ; color
7526 0000E7A7 89D1                <1>       mov    ecx, edx ; win start pos
7527 0000E7A9 89F2                <1>       mov    edx, esi ; size (rows, cols)
7528 0000E7AB E828FEFFFF          <1>       call   sysvideo_15_12 ; window preparations
7529 0000E7B0 58                  <1>       pop    eax ; * ; color
7530 0000E7B1 72F2                <1>       jc     short pix_op_add_5
7531                              <1>
7532 0000E7B3 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color adding ?
7533 0000E7BA 7405                <1>       jz     short pix_op_add_w_0 ; no
7534 0000E7BC E92A110000          <1>       jmp    m_pix_op_add_w
7535                              <1>                      ; window add color except mask color
7536                              <1> pix_op_add_w_0:
7537                              <1>       ; ecx = bytes per row (to be applied)
7538                              <1>       ; edx = screen width in bytes
7539                              <1>       ; ebx = row count
7540                              <1>       ; eax = color
7541                              <1>
7542 0000E7C1 8B3D[92120300]      <1>       mov    edi, [v_str]
7543 0000E7C7 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
```

```
7544 0000E7CE 7707               <1>       ja     short pix_op_add_w_1
7545                             <1>
7546                             <1>       ; 256 colors (8bpp)
7547 0000E7D0 BD[CAF10000]       <1>       mov    ebp, pix_op_add_8
7548 0000E7D5 EB1E               <1>       jmp    short pix_op_add_w_4
7549                             <1>
7550                             <1> pix_op_add_w_1:
7551 0000E7D7 803D[89120300]18   <1>       cmp    byte [v_bpp], 24 ; 24bpp
7552 0000E7DE 7710               <1>       ja     short pix_op_add_w_3 ; 32bpp
7553 0000E7E0 7207               <1>       jb     short pix_op_add_w_2 ; 16bpp
7554                             <1>
7555                             <1>       ; 24 bit true colors
7556 0000E7E2 BD[EAF10000]       <1>       mov    ebp, pix_op_add_24
7557 0000E7E7 EB0C               <1>       jmp    short pix_op_add_w_4
7558                             <1>
7559                             <1>       ; 65536 colors (16bpp)
7560                             <1> pix_op_add_w_2:
7561 0000E7E9 BD[D8F10000]       <1>       mov    ebp, pix_op_add_16
7562 0000E7EE EB05               <1>       jmp    short pix_op_add_w_4
7563                             <1>
7564                             <1>       ; 32 bit true colors
7565                             <1> pix_op_add_w_3:
7566 0000E7F0 BD[0AF20000]       <1>       mov    ebp, pix_op_add_32
7567                             <1> pix_op_add_w_4:
7568 0000E7F5 E95F010000         <1>       jmp    pix_op_add_w_x
7569                             <1>
7570                             <1> pix_op_sub:
7571                             <1>       ; 31/01/2021
7572                             <1>       ; SUB COLOR
7573                             <1>       ;
7574                             <1>       ; INPUT:
7575                             <1>       ;   CL = color (8 bit, 256 colors)
7576                             <1>       ;  ECX = color (16 bit and true colors)
7577                             <1>       ;  EDX = start position (row, column)
7578                             <1>       ;        (HW = row, DX = column)
7579                             <1>       ;  ESI = size (rows, colums)
7580                             <1>       ;        (HW = rows, SI = columns)
7581                             <1>       ;
7582                             <1>       ;  [maskcolor] = mask color (to be excluded)
7583                             <1>       ;
7584                             <1>       ; OUTPUT:
7585                             <1>       ;      [u.r0] will be > 0 if succesful
7586                             <1>
7587 0000E7FA F605[88120300]10   <1>       test   byte [v_ops], 10h ; display page or window ?
7588 0000E801 7555               <1>       jnz    short pix_op_sub_w ; window
7589                             <1>
7590 0000E803 8B3D[8A120300]     <1>       mov    edi, [v_mem]
7591 0000E809 89FE               <1>       mov    esi, edi
7592                             <1>       ; ecx = color (CL, CX, ECX)
7593 0000E80B 89C8               <1>       mov    eax, ecx
7594 0000E80D 8B0D[8E120300]     <1>       mov    ecx, [v_siz] ; display page pixel count
7595                             <1>
7596 0000E813 F605[88120300]20   <1>       test   byte [v_ops], 20h ; masked color subtract ?
7597 0000E81A 7405               <1>       jz     short pix_op_sub_0 ; no
7598 0000E81C E9FD100000         <1>       jmp    m_pix_op_sub ; sub color except mask color
7599                             <1> pix_op_sub_0:
7600 0000E821 803D[89120300]08   <1>       cmp    byte [v_bpp], 8 ; 8bpp
7601 0000E828 7707               <1>       ja     short pix_op_sub_1
7602                             <1>
7603                             <1>       ; 256 colors (8bpp)
7604 0000E82A E8EA090000         <1>       call   pix_op_sub_8
7605 0000E82F EB1E               <1>       jmp    short pix_op_sub_4
7606                             <1>
7607                             <1> pix_op_sub_1:
7608 0000E831 803D[89120300]18   <1>       cmp    byte [v_bpp], 24 ; 24bpp
7609 0000E838 7710               <1>       ja     short pix_op_sub_3 ; 32bpp
7610 0000E83A 7207               <1>       jb     short pix_op_sub_2 ; 16bpp
7611                             <1>
7612                             <1>       ; 24 bit true colors
7613 0000E83C E8FB090000         <1>       call   pix_op_sub_24
7614 0000E841 EB0C               <1>       jmp    short pix_op_sub_4
7615                             <1>
7616                             <1>       ; 65536 colors (16bpp)
7617                             <1> pix_op_sub_2:
7618 0000E843 E8E1090000         <1>       call   pix_op_sub_16
7619 0000E848 EB05               <1>       jmp    short pix_op_sub_4
7620                             <1>
7621                             <1>       ; 32 bit true colors
7622                             <1> pix_op_sub_3:
7623 0000E84A E8070A0000         <1>       call   pix_op_sub_32
7624                             <1> pix_op_sub_4:
7625 0000E84F 29F7               <1>       sub    edi, esi
7626 0000E851 893D[64030300]     <1>       mov    [u.r0], edi
7627                             <1> pix_op_sub_5:
7628 0000E857 C3                 <1>       retn
7629                             <1>
7630                             <1> pix_op_sub_w:
7631                             <1>       ; 31/01/2021
7632 0000E858 51                 <1>       push   ecx ; * ; color
7633 0000E859 89D1               <1>       mov    ecx, edx ; win start pos
7634 0000E85B 89F2               <1>       mov    edx, esi ; size (rows, cols)
7635 0000E85D E876FDFFFF         <1>       call   sysvideo_15_12 ; window preparations
7636 0000E862 58                 <1>       pop    eax ; * ; color
7637 0000E863 72F2               <1>       jc     short pix_op_sub_5
7638                             <1>
7639 0000E865 F605[88120300]20   <1>       test   byte [v_ops], 20h ; masked color subtract ?
7640 0000E86C 7405               <1>       jz     short pix_op_sub_w_0 ; no
7641 0000E86E E94E110000         <1>       jmp    m_pix_op_sub_w
7642                             <1>                     ; window sub color except mask color
7643                             <1> pix_op_sub_w_0:
7644                             <1>       ; ecx = bytes per row (to be applied)
7645                             <1>       ; edx = screen width in bytes
7646                             <1>       ; ebx = row count
7647                             <1>       ; eax = color
7648                             <1>
```

```
7649 0000E873 8B3D[92120300]      <1>       mov    edi, [v_str]
7650 0000E879 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7651 0000E880 7707                <1>       ja     short pix_op_sub_w_1
7652                              <1>
7653                              <1>       ; 256 colors (8bpp)
7654 0000E882 BD[19F20000]        <1>       mov    ebp, pix_op_sub_8
7655 0000E887 EB1E                <1>       jmp    short pix_op_sub_w_4
7656                              <1>
7657                              <1> pix_op_sub_w_1:
7658 0000E889 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7659 0000E890 7710                <1>       ja     short pix_op_sub_w_3 ; 32bpp
7660 0000E892 7207                <1>       jb     short pix_op_sub_w_2 ; 16bpp
7661                              <1>
7662                              <1>       ; 24 bit true colors
7663 0000E894 BD[3CF20000]        <1>       mov    ebp, pix_op_sub_24
7664 0000E899 EB0C                <1>       jmp    short pix_op_sub_w_4
7665                              <1>
7666                              <1>       ; 65536 colors (16bpp)
7667                              <1> pix_op_sub_w_2:
7668 0000E89B BD[29F20000]        <1>       mov    ebp, pix_op_sub_16
7669 0000E8A0 EB05                <1>       jmp    short pix_op_sub_w_4
7670                              <1>
7671                              <1>       ; 32 bit true colors
7672                              <1> pix_op_sub_w_3:
7673 0000E8A2 BD[56F20000]        <1>       mov    ebp, pix_op_sub_32
7674                              <1> pix_op_sub_w_4:
7675 0000E8A7 E9AD000000          <1>       jmp    pix_op_sub_w_x
7676                              <1>
7677                              <1> pix_op_mix:
7678                              <1>       ; 31/01/2021
7679                              <1>       ; MIX COLOR
7680                              <1>       ;
7681                              <1>       ; INPUT:
7682                              <1>       ;   CL = color (8 bit, 256 colors)
7683                              <1>       ;  ECX = color (16 bit and true colors)
7684                              <1>       ;  EDX = start position (row, column)
7685                              <1>       ;         (HW = row, DX = column)
7686                              <1>       ;  ESI = size (rows, colums)
7687                              <1>       ;         (HW = rows, SI = columns)
7688                              <1>       ;
7689                              <1>       ;  [maskcolor] = mask color (to be excluded)
7690                              <1>       ;
7691                              <1>       ; OUTPUT:
7692                              <1>       ;      [u.r0] will be > 0 if succesful
7693                              <1>
7694 0000E8AC F605[88120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
7695 0000E8B3 7555                <1>       jnz    short pix_op_mix_w ; window
7696                              <1>
7697 0000E8B5 8B3D[8A120300]      <1>       mov    edi, [v_mem]
7698 0000E8BB 89FE                <1>       mov    esi, edi
7699                              <1>       ; ecx = color (CL, CX, ECX)
7700 0000E8BD 89C8                <1>       mov    eax, ecx
7701 0000E8BF 8B0D[8E120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
7702                              <1>
7703 0000E8C5 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color mix ?
7704 0000E8CC 7405                <1>       jz     short pix_op_mix_0 ; no
7705 0000E8CE E921110000          <1>       jmp    m_pix_op_mix ; mix colors except mask color
7706                              <1> pix_op_mix_0:
7707 0000E8D3 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7708 0000E8DA 7707                <1>       ja     short pix_op_mix_1
7709                              <1>
7710                              <1>       ; 256 colors (8bpp)
7711 0000E8DC E8F4090000          <1>       call   pix_op_mix_8
7712 0000E8E1 EB1E                <1>       jmp    short pix_op_mix_4
7713                              <1>
7714                              <1> pix_op_mix_1:
7715 0000E8E3 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7716 0000E8EA 7710                <1>       ja     short pix_op_mix_3 ; 32bpp
7717 0000E8EC 7207                <1>       jb     short pix_op_mix_2 ; 16bpp
7718                              <1>
7719                              <1>       ; 24 bit true colors
7720 0000E8EE E8FD090000          <1>       call   pix_op_mix_24
7721 0000E8F3 EB0C                <1>       jmp    short pix_op_mix_4
7722                              <1>
7723                              <1>       ; 65536 colors (16bpp)
7724                              <1> pix_op_mix_2:
7725 0000E8F5 E8E7090000          <1>       call   pix_op_mix_16
7726 0000E8FA EB05                <1>       jmp    short pix_op_mix_4
7727                              <1>
7728                              <1>       ; 32 bit true colors
7729                              <1> pix_op_mix_3:
7730 0000E8FC E80B0A0000          <1>       call   pix_op_mix_32
7731                              <1> pix_op_mix_4:
7732 0000E901 29F7                <1>       sub    edi, esi
7733 0000E903 893D[64030300]      <1>       mov    [u.r0], edi
7734                              <1> pix_op_mix_5:
7735 0000E909 C3                  <1>       retn
7736                              <1>
7737                              <1> pix_op_mix_w:
7738                              <1>       ; 31/01/2021
7739 0000E90A 51                  <1>       push   ecx ; * ; color
7740 0000E90B 89D1                <1>       mov    ecx, edx ; win start pos
7741 0000E90D 89F2                <1>       mov    edx, esi ; size (rows, cols)
7742 0000E90F E8C4FCFFFF          <1>       call   sysvideo_15_12 ; window preparations
7743 0000E914 58                  <1>       pop    eax ; * ; color
7744 0000E915 72F2                <1>       jc     short pix_op_mix_5
7745                              <1>
7746 0000E917 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color mix ?
7747 0000E91E 7405                <1>       jz     short pix_op_mix_w_0 ; no
7748 0000E920 E96C110000          <1>       jmp    m_pix_op_mix_w
7749                              <1>                      ; window mix colors except mask color
7750                              <1> pix_op_mix_w_0:
7751                              <1>       ; ecx = bytes per row (to be applied)
7752                              <1>       ; edx = screen width in bytes
7753                              <1>       ; ebx = row count
```

```
7754                                <1>       ; eax = color
7755                                <1>
7756 0000E925 8B3D[92120300]        <1>       mov    edi, [v_str]
7757 0000E92B 803D[89120300]08      <1>       cmp    byte [v_bpp], 8 ; 8bpp
7758 0000E932 7707                  <1>       ja     short pix_op_mix_w_1
7759                                <1>
7760                                <1>       ; 256 colors (8bpp)
7761 0000E934 BD[D5F20000]          <1>       mov    ebp, pix_op_mix_8
7762 0000E939 EB1E                  <1>       jmp    short pix_op_mix_w_x
7763                                <1>
7764                                <1> pix_op_mix_w_1:
7765 0000E93B 803D[89120300]18      <1>       cmp    byte [v_bpp], 24 ; 24bpp
7766 0000E942 7710                  <1>       ja     short pix_op_mix_w_3 ; 32bpp
7767 0000E944 7207                  <1>       jb     short pix_op_mix_w_2 ; 16bpp
7768                                <1>
7769                                <1>       ; 24 bit true colors
7770 0000E946 BD[F0F20000]          <1>       mov    ebp, pix_op_mix_24
7771 0000E94B EB0C                  <1>       jmp    short pix_op_mix_w_x
7772                                <1>
7773                                <1>       ; 65536 colors (16bpp)
7774                                <1> pix_op_mix_w_2:
7775 0000E94D BD[E1F20000]          <1>       mov    ebp, pix_op_mix_16
7776 0000E952 EB05                  <1>       jmp    short pix_op_mix_w_x
7777                                <1>
7778                                <1>       ; 32 bit true colors
7779                                <1> pix_op_mix_w_3:
7780 0000E954 BD[0CF30000]          <1>       mov    ebp, pix_op_mix_32
7781                                <1>       ;jmp   short pix_op_mix_w_x
7782                                <1>
7783                                <1> pix_op_mix_w_x:
7784                                <1> pix_op_add_w_x:
7785                                <1> pix_op_sub_w_x:
7786                                <1> pix_op_rpl_w_x:
7787                                <1> pix_op_orc_w_x:
7788                                <1> pix_op_and_w_x:
7789                                <1> pix_op_xor_w_x:
7790                                <1>       ; 27/02/2021
7791                                <1>       ; 31/01/2021
7792                                <1>       ; ecx = bytes per row (to be applied)
7793                                <1>       ; edx = windows (screen) width in bytes
7794                                <1>       ; ebx = row count
7795                                <1>       ; eax = color
7796                                <1>       ; ebp = pixel operation subroutine address
7797 0000E959 52                    <1>       push   edx
7798 0000E95A 51                    <1>       push   ecx
7799 0000E95B 57                    <1>       push   edi
7800 0000E95C 8B0D[9E120300]        <1>       mov    ecx, [pixcount] ; 27/02/2021
7801 0000E962 FFD5                  <1>       call   ebp ; call pixel-row operation
7802 0000E964 5F                    <1>       pop    edi
7803 0000E965 59                    <1>       pop    ecx ; bytes per row
7804 0000E966 010D[64030300]        <1>       add    [u.r0], ecx
7805 0000E96C 5A                    <1>       pop    edx
7806 0000E96D 01D7                  <1>       add    edi, edx ; next row
7807 0000E96F 4B                    <1>       dec    ebx
7808 0000E970 75E7                  <1>       jnz    short pix_op_mix_w_x
7809 0000E972 C3                    <1>       retn
7810                                <1>
7811                                <1> pix_op_rpl:
7812                                <1>       ; 01/02/2021
7813                                <1>       ; REPLACE COLOR
7814                                <1>       ;
7815                                <1>       ; INPUT:
7816                                <1>       ;   CL = old/current color (8 bit, 256 colors)
7817                                <1>       ;  ECX = old/current color (16 bit and true colors)
7818                                <1>       ;   DL = new color (8 bit, 256 colors)
7819                                <1>       ;  EDX = new color (16 bit and true colors)
7820                                <1>       ;  ESI = start position (row, column)
7821                                <1>       ;        (HW = row, DX = column)
7822                                <1>       ;  EDI = size (rows, colums)
7823                                <1>       ;        (HW = rows, SI = columns)
7824                                <1>       ; OUTPUT:
7825                                <1>       ;     [u.r0] will be > 0 if succesful
7826                                <1>
7827 0000E973 F605[88120300]10      <1>       test   byte [v_ops], 10h ; display page or window ?
7828 0000E97A 754D                  <1>       jnz    short pix_op_rpl_w ; window
7829                                <1>
7830 0000E97C 8B3D[8A120300]        <1>       mov    edi, [v_mem]
7831 0000E982 89FE                  <1>       mov    esi, edi
7832                                <1>       ; ecx = old color (CL, CX, ECX) -to be replaced with-
7833                                <1>       ; edx = new color (CL, CX, ECX) -new one-
7834 0000E984 89D0                  <1>       mov    eax, edx ; new color
7835 0000E986 890D[9A120300]        <1>       mov    [maskcolor], ecx ; old color
7836 0000E98C 8B0D[8E120300]        <1>       mov    ecx, [v_siz] ; display page pixel count
7837                                <1> pix_op_rpl_0:
7838 0000E992 803D[89120300]08      <1>       cmp    byte [v_bpp], 8 ; 8bpp
7839 0000E999 7707                  <1>       ja     short pix_op_rpl_1
7840                                <1>
7841                                <1>       ; 256 colors (8bpp)
7842 0000E99B E8300A0000            <1>       call   pix_op_rpl_8
7843 0000E9A0 EB1E                  <1>       jmp    short pix_op_rpl_4
7844                                <1>
7845                                <1> pix_op_rpl_1:
7846 0000E9A2 803D[89120300]18      <1>       cmp    byte [v_bpp], 24 ; 24bpp
7847 0000E9A9 7710                  <1>       ja     short pix_op_rpl_3 ; 32bpp
7848 0000E9AB 7207                  <1>       jb     short pix_op_rpl_2 ; 16bpp
7849                                <1>
7850                                <1>       ; 24 bit true colors
7851 0000E9AD E8410A0000            <1>       call   pix_op_rpl_24
7852 0000E9B2 EB0C                  <1>       jmp    short pix_op_rpl_4
7853                                <1>
7854                                <1>       ; 65536 colors (16bpp)
7855                                <1> pix_op_rpl_2:
7856 0000E9B4 E8270A0000            <1>       call   pix_op_rpl_16
7857 0000E9B9 EB05                  <1>       jmp    short pix_op_rpl_4
7858                                <1>
```

```
                            <1>       ; 32 bit true colors
                            <1> pix_op_rpl_3:
0000E9BB E8550A0000         <1>       call   pix_op_rpl_32
                            <1> pix_op_rpl_4:
0000E9C0 29F7               <1>       sub    edi, esi
0000E9C2 893D[64030300]     <1>       mov    [u.r0], edi
                            <1> pix_op_rpl_5:
0000E9C8 C3                 <1>       retn
                            <1>
                            <1> pix_op_rpl_w:
                            <1>       ; 01/02/2021
0000E9C9 890D[9A120300]     <1>       mov    [maskcolor], ecx ; old color
0000E9CF 52                 <1>       push   edx ; * ; new color
0000E9D0 89F1               <1>       mov    ecx, esi ; win start pos
0000E9D2 89FA               <1>       mov    edx, edi ; size (rows, cols)
0000E9D4 E8FFFBFFFF         <1>       call   sysvideo_15_12 ; window preparations
0000E9D9 58                 <1>       pop    eax ; * ; new color
0000E9DA 72EC               <1>       jc     short pix_op_rpl_5
                            <1>
                            <1>       ; replace window color
                            <1> pix_op_rpl_w_0:
                            <1>       ; ecx = bytes per row (to be applied)
                            <1>       ; edx = screen width in bytes
                            <1>       ; ebx = row count
                            <1>       ; eax = new color
                            <1>       ; [maskcolor] = old color
                            <1>
0000E9DC 8B3D[92120300]     <1>       mov    edi, [v_str]
                            <1>
0000E9E2 803D[89120300]08   <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E9E9 7707               <1>       ja     short pix_op_rpl_w_1
                            <1>
                            <1>       ; 256 colors (8bpp)
0000E9EB BD[D0F30000]       <1>       mov    ebp, pix_op_rpl_8
0000E9F0 EB1E               <1>       jmp    short pix_op_rpl_w_4
                            <1>
                            <1> pix_op_rpl_w_1:
0000E9F2 803D[89120300]18   <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E9F9 7710               <1>       ja     short pix_op_rpl_w_3 ; 32bpp
0000E9FB 7207               <1>       jb     short pix_op_rpl_w_2 ; 16bpp
                            <1>
                            <1>       ; 24 bit true colors
0000E9FD BD[F3F30000]       <1>       mov    ebp, pix_op_rpl_24
0000EA02 EB0C               <1>       jmp    short pix_op_rpl_w_4
                            <1>
                            <1>       ; 65536 colors (16bpp)
                            <1> pix_op_rpl_w_2:
0000EA04 BD[E0F30000]       <1>       mov    ebp, pix_op_rpl_16
0000EA09 EB05               <1>       jmp    short pix_op_rpl_w_4
                            <1>
                            <1>       ; 32 bit true colors
                            <1> pix_op_rpl_w_3:
0000EA0B BD[15F40000]       <1>       mov    ebp, pix_op_rpl_32
                            <1> pix_op_rpl_w_4:
0000EA10 E944FFFFFF         <1>       jmp    pix_op_rpl_w_x
                            <1>
                            <1> pix_op_orc:
                            <1>       ; 31/01/2021
                            <1>       ; OR COLOR
                            <1>       ;
                            <1>       ; INPUT:
                            <1>       ;   CL = color (8 bit, 256 colors)
                            <1>       ;  ECX = color (16 bit and true colors)
                            <1>       ;  EDX = start position (row, column)
                            <1>       ;        (HW = row, DX = column)
                            <1>       ;  ESI = size (rows, colums)
                            <1>       ;        (HW = rows, SI = columns)
                            <1>       ;
                            <1>       ;  [maskcolor] = mask color (to be excluded)
                            <1>       ;
                            <1>       ; OUTPUT:
                            <1>       ;      [u.r0] will be > 0 if succesful
                            <1>
0000EA15 F605[88120300]10   <1>       test   byte [v_ops], 10h ; display page or window ?
0000EA1C 7555               <1>       jnz    short pix_op_or_w ; window
                            <1>
0000EA1E 8B3D[8A120300]     <1>       mov    edi, [v_mem]
0000EA24 89FE               <1>       mov    esi, edi
                            <1>       ; ecx = color (CL, CX, ECX)
0000EA26 89C8               <1>       mov    eax, ecx
0000EA28 8B0D[8E120300]     <1>       mov    ecx, [v_siz] ; display page pixel count
                            <1>
0000EA2E F605[88120300]20   <1>       test   byte [v_ops], 20h ; masked color 'or' ?
0000EA35 7405               <1>       jz     short pix_op_or_0 ; no
0000EA37 E948110000         <1>       jmp    m_pix_op_or ; 'or' color except mask color
                            <1> pix_op_or_0:
0000EA3C 803D[89120300]08   <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EA43 7707               <1>       ja     short pix_op_or_1
                            <1>
                            <1>       ; 256 colors (8bpp)
0000EA45 E81C080000         <1>       call   pix_op_or_8
0000EA4A EB1E               <1>       jmp    short pix_op_or_4
                            <1>
                            <1> pix_op_or_1:
0000EA4C 803D[89120300]18   <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EA53 7710               <1>       ja     short pix_op_or_3 ; 32bpp
0000EA55 7207               <1>       jb     short pix_op_or_2 ; 16bpp
                            <1>
                            <1>       ; 24 bit true colors
0000EA57 E818080000         <1>       call   pix_op_or_24
0000EA5C EB0C               <1>       jmp    short pix_op_or_4
                            <1>
                            <1>       ; 65536 colors (16bpp)
                            <1> pix_op_or_2:
0000EA5E E809080000         <1>       call   pix_op_or_16
```

```
7964 0000EA63 EB05                <1>       jmp    short pix_op_or_4
7965                              <1>
7966                              <1>       ; 32 bit true colors
7967                              <1> pix_op_or_3:
7968 0000EA65 E819080000          <1>       call   pix_op_or_32
7969                              <1> pix_op_or_4:
7970 0000EA6A 29F7                <1>       sub    edi, esi
7971 0000EA6C 893D[64030300]      <1>       mov    [u.r0], edi
7972                              <1> pix_op_or_5:
7973 0000EA72 C3                  <1>       retn
7974                              <1>
7975                              <1> pix_op_or_w:
7976                              <1>       ; 31/01/2021
7977 0000EA73 51                  <1>       push   ecx ; * ; color
7978 0000EA74 89D1                <1>       mov    ecx, edx ; win start pos
7979 0000EA76 89F2                <1>       mov    edx, esi ; size (rows, cols)
7980 0000EA78 E85BFBFFFF          <1>       call   sysvideo_15_12 ; window preparations
7981 0000EA7D 58                  <1>       pop    eax ; * ; color
7982 0000EA7E 72F2                <1>       jc     short pix_op_or_5
7983                              <1>
7984 0000EA80 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color 'or' ?
7985 0000EA87 7405                <1>       jz     short pix_op_or_w_0 ; no
7986 0000EA89 E983110000          <1>       jmp    m_pix_op_or_w
7987                              <1>                     ; window 'or' color except mask color
7988                              <1> pix_op_or_w_0:
7989                              <1>       ; ecx = bytes per row (to be applied)
7990                              <1>       ; edx = screen width in bytes
7991                              <1>       ; ebx = row count
7992                              <1>       ; eax = color
7993                              <1>
7994 0000EA8E 8B3D[92120300]      <1>       mov    edi, [v_str]
7995 0000EA94 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7996 0000EA9B 7707                <1>       ja     short pix_op_or_w_1
7997                              <1>
7998                              <1>       ; 256 colors (8bpp)
7999 0000EA9D BD[66F20000]        <1>       mov    ebp, pix_op_or_8
8000 0000EAA2 EB1E                <1>       jmp    short pix_op_or_w_4
8001                              <1>
8002                              <1> pix_op_or_w_1:
8003 0000EAA4 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8004 0000EAAB 7710                <1>       ja     short pix_op_or_w_3 ; 32bpp
8005 0000EAAD 7207                <1>       jb     short pix_op_or_w_2 ; 16bpp
8006                              <1>
8007                              <1>       ; 24 bit true colors
8008 0000EAAF BD[74F20000]        <1>       mov    ebp, pix_op_or_24
8009 0000EAB4 EB0C                <1>       jmp    short pix_op_or_w_4
8010                              <1>
8011                              <1>       ; 65536 colors (16bpp)
8012                              <1> pix_op_or_w_2:
8013 0000EAB6 BD[6CF20000]        <1>       mov    ebp, pix_op_or_16
8014 0000EABB EB05                <1>       jmp    short pix_op_or_w_4
8015                              <1>
8016                              <1>       ; 32 bit true colors
8017                              <1> pix_op_or_w_3:
8018 0000EABD BD[83F20000]        <1>       mov    ebp, pix_op_or_32
8019                              <1> pix_op_or_w_4:
8020 0000EAC2 E992FEFFFF          <1>       jmp    pix_op_orc_w_x
8021                              <1>
8022                              <1> pix_op_and:
8023                              <1>       ; 31/01/2021
8024                              <1>       ; AND COLOR
8025                              <1>       ;
8026                              <1>       ; INPUT:
8027                              <1>       ;   CL = color (8 bit, 256 colors)
8028                              <1>       ;  ECX = color (16 bit and true colors)
8029                              <1>       ;  EDX = start position (row, column)
8030                              <1>       ;        (HW = row, DX = column)
8031                              <1>       ;  ESI = size (rows, colums)
8032                              <1>       ;        (HW = rows, SI = columns)
8033                              <1>       ;
8034                              <1>       ;  [maskcolor] = mask color (to be excluded)
8035                              <1>       ;
8036                              <1>       ; OUTPUT:
8037                              <1>       ;      [u.r0] will be > 0 if succesful
8038                              <1>
8039 0000EAC7 F605[88120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
8040 0000EACE 7555                <1>       jnz    short pix_op_and_w ; window
8041                              <1>
8042 0000EAD0 8B3D[8A120300]      <1>       mov    edi, [v_mem]
8043 0000EAD6 89FE                <1>       mov    esi, edi
8044                              <1>       ; ecx = color (CL, CX, ECX)
8045 0000EAD8 89C8                <1>       mov    eax, ecx
8046 0000EADA 8B0D[8E120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
8047                              <1>
8048 0000EAE0 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color 'and' ?
8049 0000EAE7 7405                <1>       jz     short pix_op_and_0 ; no
8050 0000EAE9 E9D60F0000          <1>       jmp    m_pix_op_and ; 'and' color except mask color
8051                              <1> pix_op_and_0:
8052 0000EAEE 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8053 0000EAF5 7707                <1>       ja     short pix_op_and_1
8054                              <1>
8055                              <1>       ; 256 colors (8bpp)
8056 0000EAF7 E88F070000          <1>       call   pix_op_and_8
8057 0000EAFC EB1E                <1>       jmp    short pix_op_and_4
8058                              <1>
8059                              <1> pix_op_and_1:
8060 0000EAFE 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8061 0000EB05 7710                <1>       ja     short pix_op_and_3 ; 32bpp
8062 0000EB07 7207                <1>       jb     short pix_op_and_2 ; 16bpp
8063                              <1>
8064                              <1>       ; 24 bit true colors
8065 0000EB09 E88B070000          <1>       call   pix_op_and_24
8066 0000EB0E EB0C                <1>       jmp    short pix_op_and_4
8067                              <1>
8068                              <1>       ; 65536 colors (16bpp)
```

```
                                <1> pix_op_and_2:
0000EB10 E87C070000             <1>         call    pix_op_and_16
0000EB15 EB05                   <1>         jmp     short pix_op_and_4
                                <1>
                                <1>         ; 32 bit true colors
                                <1> pix_op_and_3:
0000EB17 E88C070000             <1>         call    pix_op_and_32
                                <1> pix_op_and_4:
0000EB1C 29F7                   <1>         sub     edi, esi
0000EB1E 893D[64030300]         <1>         mov     [u.r0], edi
                                <1> pix_op_and_5:
0000EB24 C3                     <1>         retn
                                <1>
                                <1> pix_op_and_w:
                                <1>         ; 31/01/2021
0000EB25 51                     <1>         push    ecx ; * ; color
0000EB26 89D1                   <1>         mov     ecx, edx ; win start pos
0000EB28 89F2                   <1>         mov     edx, esi ; size (rows, cols)
0000EB2A E8A9FAFFFF             <1>         call    sysvideo_15_12 ; window preparations
0000EB2F 58                     <1>         pop     eax ; * ; color
0000EB30 72F2                   <1>         jc      short pix_op_and_5
                                <1>
0000EB32 F605[88120300]20       <1>         test    byte [v_ops], 20h ; masked color 'and' ?
0000EB39 7405                   <1>         jz      short pix_op_and_w_0 ; no
0000EB3B E911100000             <1>         jmp     m_pix_op_and_w
                                <1>                         ; window 'and' color except mask color
                                <1> pix_op_and_w_0:
                                <1>         ; ecx = bytes per row (to be applied)
                                <1>         ; edx = screen width in bytes
                                <1>         ; ebx = row count
                                <1>         ; eax = color
                                <1>
0000EB40 8B3D[92120300]         <1>         mov     edi, [v_str]
0000EB46 803D[89120300]08       <1>         cmp     byte [v_bpp], 8 ; 8bpp
0000EB4D 7707                   <1>         ja      short pix_op_and_w_1
                                <1>
                                <1>         ; 256 colors (8bpp)
0000EB4F BD[8BF20000]           <1>         mov     ebp, pix_op_and_8
0000EB54 EB1E                   <1>         jmp     short pix_op_and_w_4
                                <1>
                                <1> pix_op_and_w_1:
0000EB56 803D[89120300]18       <1>         cmp     byte [v_bpp], 24 ; 24bpp
0000EB5D 7710                   <1>         ja      short pix_op_and_w_3 ; 32bpp
0000EB5F 7207                   <1>         jb      short pix_op_and_w_2 ; 16bpp
                                <1>
                                <1>         ; 24 bit true colors
0000EB61 BD[99F20000]           <1>         mov     ebp, pix_op_and_24
0000EB66 EB0C                   <1>         jmp     short pix_op_and_w_4
                                <1>
                                <1>         ; 65536 colors (16bpp)
                                <1> pix_op_and_w_2:
0000EB68 BD[91F20000]           <1>         mov     ebp, pix_op_and_16
0000EB6D EB05                   <1>         jmp     short pix_op_and_w_4
                                <1>
                                <1>         ; 32 bit true colors
                                <1> pix_op_and_w_3:
0000EB6F BD[A8F20000]           <1>         mov     ebp, pix_op_and_32
                                <1> pix_op_and_w_4:
0000EB74 E9E0FDFFFF             <1>         jmp     pix_op_and_w_x
                                <1>
                                <1> pix_op_xor:
                                <1>         ; 31/01/2021
                                <1>         ; XOR COLOR
                                <1>         ;
                                <1>         ; INPUT:
                                <1>         ;   CL = color (8 bit, 256 colors)
                                <1>         ;  ECX = color (16 bit and true colors)
                                <1>         ;  EDX = start position (row, column)
                                <1>         ;        (HW = row, DX = column)
                                <1>         ;  ESI = size (rows, colums)
                                <1>         ;        (HW = rows, SI = columns)
                                <1>         ;
                                <1>         ;  [maskcolor] = mask color (to be excluded)
                                <1>         ;
                                <1>         ; OUTPUT:
                                <1>         ;       [u.r0] will be > 0 if succesful
                                <1>
0000EB79 F605[88120300]10       <1>         test    byte [v_ops], 10h ; display page or window ?
0000EB80 7555                   <1>         jnz     short pix_op_xor_w ; window
                                <1>
0000EB82 8B3D[8A120300]         <1>         mov     edi, [v_mem]
0000EB88 89FE                   <1>         mov     esi, edi
                                <1>         ; ecx = color (CL, CX, ECX)
0000EB8A 89C8                   <1>         mov     eax, ecx
0000EB8C 8B0D[8E120300]         <1>         mov     ecx, [v_siz] ; display page pixel count
                                <1>
0000EB92 F605[88120300]20       <1>         test    byte [v_ops], 20h ; masked color 'xor' ?
0000EB99 7405                   <1>         jz      short pix_op_xor_0 ; no
0000EB9B E9A4100000             <1>         jmp     m_pix_op_xor ; 'xor' color except mask color
                                <1> pix_op_xor_0:
0000EBA0 803D[89120300]08       <1>         cmp     byte [v_bpp], 8 ; 8bpp
0000EBA7 7707                   <1>         ja      short pix_op_xor_1
                                <1>
                                <1>         ; 256 colors (8bpp)
0000EBA9 E802070000             <1>         call    pix_op_xor_8
0000EBAE EB1E                   <1>         jmp     short pix_op_xor_4
                                <1>
                                <1> pix_op_xor_1:
0000EBB0 803D[89120300]18       <1>         cmp     byte [v_bpp], 24 ; 24bpp
0000EBB7 7710                   <1>         ja      short pix_op_xor_3 ; 32bpp
0000EBB9 7207                   <1>         jb      short pix_op_xor_2 ; 16bpp
                                <1>
                                <1>         ; 24 bit true colors
0000EBBB E8FE060000             <1>         call    pix_op_xor_24
0000EBC0 EB0C                   <1>         jmp     short pix_op_xor_4
```

```
8174                                    <1>
8175                                    <1>         ; 65536 colors (16bpp)
8176                                    <1> pix_op_xor_2:
8177 0000EBC2 E8EF060000                <1>         call    pix_op_xor_16
8178 0000EBC7 EB05                      <1>         jmp     short pix_op_xor_4
8179                                    <1>
8180                                    <1>         ; 32 bit true colors
8181                                    <1> pix_op_xor_3:
8182 0000EBC9 E8FF060000                <1>         call    pix_op_xor_32
8183                                    <1> pix_op_xor_4:
8184 0000EBCE 29F7                      <1>         sub     edi, esi
8185 0000EBD0 893D[64030300]            <1>         mov     [u.r0], edi
8186                                    <1> pix_op_xor_5:
8187 0000EBD6 C3                        <1>         retn
8188                                    <1>
8189                                    <1> pix_op_xor_w:
8190                                    <1>         ; 31/01/2021
8191 0000EBD7 51                        <1>         push    ecx ; * ; color
8192 0000EBD8 89D1                      <1>         mov     ecx, edx ; win start pos
8193 0000EBDA 89F2                      <1>         mov     edx, esi ; size (rows, cols)
8194 0000EBDC E8F7F9FFFF                <1>         call    sysvideo_15_12 ; window preparations
8195 0000EBE1 58                        <1>         pop     eax ; * ; color
8196 0000EBE2 72F2                      <1>         jc      short pix_op_xor_5
8197                                    <1>
8198 0000EBE4 F605[88120300]20          <1>         test    byte [v_ops], 20h ; masked color 'xor' ?
8199 0000EBEB 7405                      <1>         jz      short pix_op_xor_w_0 ; no
8200 0000EBED E9DF100000                <1>         jmp     m_pix_op_xor_w
8201                                    <1>                         ; window 'xor' color except mask color
8202                                    <1> pix_op_xor_w_0:
8203                                    <1>         ; ecx = bytes per row (to be applied)
8204                                    <1>         ; edx = screen width in bytes
8205                                    <1>         ; ebx = row count
8206                                    <1>         ; eax = color
8207                                    <1>
8208 0000EBF2 8B3D[92120300]            <1>         mov     edi, [v_str]
8209 0000EBF8 803D[89120300]08          <1>         cmp     byte [v_bpp], 8 ; 8bpp
8210 0000EBFF 7707                      <1>         ja      short pix_op_xor_w_1
8211                                    <1>
8212                                    <1>         ; 256 colors (8bpp)
8213 0000EC01 BD[B0F20000]              <1>         mov     ebp, pix_op_xor_8
8214 0000EC06 EB1E                      <1>         jmp     short pix_op_xor_w_4
8215                                    <1>
8216                                    <1> pix_op_xor_w_1:
8217 0000EC08 803D[89120300]18          <1>         cmp     byte [v_bpp], 24 ; 24bpp
8218 0000EC0F 7710                      <1>         ja      short pix_op_xor_w_3 ; 32bpp
8219 0000EC11 7207                      <1>         jb      short pix_op_xor_w_2 ; 16bpp
8220                                    <1>
8221                                    <1>         ; 24 bit true colors
8222 0000EC13 BD[BEF20000]              <1>         mov     ebp, pix_op_xor_24
8223 0000EC18 EB0C                      <1>         jmp     short pix_op_xor_w_4
8224                                    <1>
8225                                    <1>         ; 65536 colors (16bpp)
8226                                    <1> pix_op_xor_w_2:
8227 0000EC1A BD[B6F20000]              <1>         mov     ebp, pix_op_xor_16
8228 0000EC1F EB05                      <1>         jmp     short pix_op_xor_w_4
8229                                    <1>
8230                                    <1>         ; 32 bit true colors
8231                                    <1> pix_op_xor_w_3:
8232 0000EC21 BD[CDF20000]              <1>         mov     ebp, pix_op_xor_32
8233                                    <1> pix_op_xor_w_4:
8234 0000EC26 E92EFDFFFF                <1>         jmp     pix_op_xor_w_x
8235                                    <1>
8236                                    <1> pix_op_new:
8237                                    <1>         ; 31/01/2021
8238                                    <1>         ; 30/01/2021
8239                                    <1>         ; CHANGE COLOR
8240                                    <1>         ;
8241                                    <1>         ; INPUT:
8242                                    <1>         ;   CL = color (8 bit, 256 colors)
8243                                    <1>         ;  ECX = color (16 bit and true colors)
8244                                    <1>         ;  EDX = start position (row, column)
8245                                    <1>         ;         (HW = row, DX = column)
8246                                    <1>         ;  ESI = size (rows, colums)
8247                                    <1>         ;         (HW = rows, SI = columns)
8248                                    <1>         ;
8249                                    <1>         ;  [maskcolor] = mask color (to be excluded)
8250                                    <1>         ;
8251                                    <1>         ; OUTPUT:
8252                                    <1>         ;       [u.r0] will be > 0 if succesful
8253                                    <1>
8254 0000EC2B F605[88120300]10          <1>         test    byte [v_ops], 10h ; display page or window ?
8255 0000EC32 7554                      <1>         jnz     short pix_op_new_w ; window
8256                                    <1>
8257 0000EC34 8B3D[8A120300]            <1>         mov     edi, [v_mem]
8258 0000EC3A 89FE                      <1>         mov     esi, edi
8259                                    <1>         ; ecx = color (CL, CX, ECX)
8260 0000EC3C 89C8                      <1>         mov     eax, ecx
8261 0000EC3E 8B0D[8E120300]            <1>         mov     ecx, [v_siz] ; display page pixel count
8262                                    <1>
8263 0000EC44 F605[88120300]20          <1>         test    byte [v_ops], 20h ; masked color change ?
8264 0000EC4B 7405                      <1>         jz      short pix_op_new_0 ; no
8265 0000EC4D E90D0B0000                <1>         jmp     m_pix_op_new ; change color except mask color
8266                                    <1> pix_op_new_0:
8267 0000EC52 803D[89120300]08          <1>         cmp     byte [v_bpp], 8 ; 8bpp
8268 0000EC59 7706                      <1>         ja      short pix_op_new_2
8269                                    <1>
8270                                    <1>         ; 256 colors (8bpp)
8271                                    <1> pix_op_new_1:
8272 0000EC5B 88C4                      <1>         mov     ah, al
8273 0000EC5D D1E9                      <1>         shr     ecx, 1
8274 0000EC5F EB12                      <1>         jmp     short pix_op_new_3
8275                                    <1>
8276                                    <1> pix_op_new_2:
8277 0000EC61 803D[89120300]18          <1>         cmp     byte [v_bpp], 24 ; 24bpp
8278 0000EC68 7713                      <1>         ja      short pix_op_new_4 ; 32bpp
```

```
8279 0000EC6A 7207              <1>       jb     short pix_op_new_3 ; 16bpp
8280                            <1>
8281                            <1>       ; 31/01/2021
8282                            <1>
8283                            <1>       ; 24 bit true colors
8284 0000EC6C E84A050000        <1>       call   pix_op_new_24
8285                            <1>
8286 0000EC71 EB0C              <1>       jmp    short pix_op_new_5
8287                            <1>
8288                            <1>       ; 65536 colors (16bpp)
8289                            <1> pix_op_new_3:
8290 0000EC73 89C2              <1>       mov    edx, eax
8291 0000EC75 C1E010            <1>       shl    eax, 16
8292 0000EC78 6689D0            <1>       mov    ax, dx
8293 0000EC7B D1E9              <1>       shr    ecx, 1 ; dword counts
8294                            <1>       ; 32 bit true colors
8295                            <1> pix_op_new_4:
8296 0000EC7D F3AB              <1>       rep    stosd
8297                            <1> pix_op_new_5:
8298 0000EC7F 29F7              <1>       sub    edi, esi
8299 0000EC81 893D[64030300]    <1>       mov    [u.r0], edi
8300                            <1> pix_op_new_6:
8301 0000EC87 C3                <1>       retn
8302                            <1>
8303                            <1> pix_op_new_w:
8304                            <1>       ; 31/01/2021
8305                            <1>       ; 30/01/2021
8306 0000EC88 51                <1>       push   ecx ; * ; color
8307 0000EC89 89D1              <1>       mov    ecx, edx ; win start pos
8308 0000EC8B 89F2              <1>       mov    edx, esi ; size (rows, cols)
8309 0000EC8D E846F9FFFF        <1>       call   sysvideo_15_12 ; window preparations
8310 0000EC92 58                <1>       pop    eax ; * ; color
8311 0000EC93 72F2              <1>       jc     short pix_op_new_6
8312                            <1>
8313 0000EC95 F605[88120300]20  <1>       test   byte [v_ops], 20h ; masked color change ?
8314 0000EC9C 7405              <1>       jz     short pix_op_new_w_0 ; no
8315 0000EC9E E94A0B0000        <1>       jmp    m_pix_op_new_w
8316                            <1>                    ; window chg color except mask color
8317                            <1> pix_op_new_w_0:
8318                            <1>       ; ecx = bytes per row (to be applied)
8319                            <1>       ; edx = screen width in bytes
8320                            <1>       ; ebx = row count
8321                            <1>       ; eax = color
8322                            <1>
8323 0000ECA3 8B3D[92120300]    <1>       mov    edi, [v_str]
8324                            <1>
8325 0000ECA9 803D[89120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
8326 0000ECB0 7707              <1>       ja     short pix_op_new_w_1
8327                            <1>
8328                            <1>       ; 256 colors (8bpp)
8329 0000ECB2 BD[B4F10000]      <1>       mov    ebp, pix_op_new_8
8330 0000ECB7 EB1E              <1>       jmp    short pix_op_new_w_x
8331                            <1>
8332                            <1> pix_op_new_w_1:
8333 0000ECB9 803D[89120300]18  <1>       cmp    byte [v_bpp], 24 ; 24bpp
8334 0000ECC0 7710              <1>       ja     short pix_op_new_w_3 ; 32bpp
8335 0000ECC2 7207              <1>       jb     short pix_op_new_w_2 ; 16bpp
8336                            <1>
8337                            <1>       ; 24 bit true colors
8338 0000ECC4 BD[BBF10000]      <1>       mov    ebp, pix_op_new_24
8339 0000ECC9 EB0C              <1>       jmp    short pix_op_new_w_x
8340                            <1>
8341                            <1>       ; 65536 colors (16bpp)
8342                            <1> pix_op_new_w_2:
8343 0000ECCB BD[B7F10000]      <1>       mov    ebp, pix_op_new_16
8344 0000ECD0 EB05              <1>       jmp    short pix_op_new_w_x
8345                            <1>
8346                            <1>       ; 32 bit true colors
8347                            <1> pix_op_new_w_3:
8348 0000ECD2 BD[C7F10000]      <1>       mov    ebp, pix_op_new_32
8349                            <1>       ;jmp   short pix_op_new_w_x
8350                            <1>
8351                            <1> pix_op_new_w_x:
8352                            <1> pix_op_not_w_x:
8353                            <1> pix_op_neg_w_x:
8354                            <1> pix_op_inc_w_x:
8355                            <1> pix_op_dec_w_x:
8356                            <1>       ; 27/02/2021
8357                            <1>       ; 01/02/2021
8358                            <1>       ; 31/01/2021
8359                            <1>       ; ecx = bytes per row (to be applied)
8360                            <1>       ; edx = windows (screen) width in bytes
8361                            <1>       ; ebx = row count
8362                            <1>       ; eax = color
8363                            <1>       ; ebp = pixel operation subroutine address
8364                            <1>       ;push  edx ; 01/02/2021
8365 0000ECD7 51                <1>       push   ecx
8366 0000ECD8 57                <1>       push   edi
8367 0000ECD9 8B0D[9E120300]    <1>       mov    ecx, [pixcount] ; 27/02/2021
8368 0000ECDF FFD5              <1>       call   ebp ; call pixel-row operation
8369 0000ECE1 5F                <1>       pop    edi
8370 0000ECE2 59                <1>       pop    ecx ; bytes per row
8371 0000ECE3 010D[64030300]    <1>       add    [u.r0], ecx
8372                            <1>       ;pop   edx ; 01/02/2021
8373 0000ECE9 01D7              <1>       add    edi, edx ; next row
8374 0000ECEB 4B                <1>       dec    ebx
8375 0000ECEC 75E9              <1>       jnz    short pix_op_new_w_x
8376 0000ECEE C3                <1>       retn
8377                            <1>
8378                            <1> pix_op_not:
8379                            <1>       ; 31/01/2021
8380                            <1>       ; NOT COLOR
8381                            <1>       ;
8382                            <1>       ; INPUT:
8383                            <1>       ;  ECX = start position (row, column)
```

```
8384                                    <1>       ;         (HW = row, CX = column)
8385                                    <1>       ;  EDX = size (rows, colums)
8386                                    <1>       ;         (HW = rows, DX = columns)
8387                                    <1>       ;       (0 -> invalid
8388                                    <1>       ;         (1 -> horizontal or vertical line)
8389                                    <1>       ;  [maskcolor] = mask color (to be excluded)
8390                                    <1>       ;
8391                                    <1>       ; OUTPUT:
8392                                    <1>       ;      [u.r0] will be > 0 if succesful
8393                                    <1>
8394 0000ECEF F605[88120300]10         <1>       test   byte [v_ops], 10h ; display page or window ?
8395 0000ECF6 7553                     <1>       jnz    short pix_op_not_w ; window
8396                                    <1>
8397 0000ECF8 8B3D[8A120300]           <1>       mov    edi, [v_mem]
8398 0000ECFE 89FE                     <1>       mov    esi, edi
8399 0000ED00 8B0D[8E120300]           <1>       mov    ecx, [v_siz] ; display page pixel count
8400                                    <1>
8401 0000ED06 F605[88120300]20         <1>       test   byte [v_ops], 20h ; masked color 'not' ?
8402 0000ED0D 7405                     <1>       jz     short pix_op_not_0 ; no
8403 0000ED0F E9F00F0000               <1>       jmp    m_pix_op_not ; 'not' color except mask color
8404                                    <1> pix_op_not_0:
8405 0000ED14 803D[89120300]08         <1>       cmp    byte [v_bpp], 8 ; 8bpp
8406 0000ED1B 7707                     <1>       ja     short pix_op_not_1
8407                                    <1>
8408                                    <1>       ; 256 colors (8bpp)
8409 0000ED1D E8F6050000               <1>       call   pix_op_not_8
8410 0000ED22 EB1E                     <1>       jmp    short pix_op_not_4
8411                                    <1>
8412                                    <1> pix_op_not_1:
8413 0000ED24 803D[89120300]18         <1>       cmp    byte [v_bpp], 24 ; 24bpp
8414 0000ED2B 7710                     <1>       ja     short pix_op_not_3 ; 32bpp
8415 0000ED2D 7207                     <1>       jb     short pix_op_not_2 ; 16bpp
8416                                    <1>
8417                                    <1>       ; 24 bit true colors
8418 0000ED2F E8F2050000               <1>       call   pix_op_not_24
8419 0000ED34 EB0C                     <1>       jmp    short pix_op_not_4
8420                                    <1>
8421                                    <1>       ; 65536 colors (16bpp)
8422                                    <1> pix_op_not_2:
8423 0000ED36 E8E3050000               <1>       call   pix_op_not_16
8424 0000ED3B EB05                     <1>       jmp    short pix_op_not_4
8425                                    <1>
8426                                    <1>       ; 32 bit true colors
8427                                    <1> pix_op_not_3:
8428 0000ED3D E8EF050000               <1>       call   pix_op_not_32
8429                                    <1> pix_op_not_4:
8430 0000ED42 29F7                     <1>       sub    edi, esi
8431 0000ED44 893D[64030300]           <1>       mov    [u.r0], edi
8432                                    <1> pix_op_not_5:
8433 0000ED4A C3                       <1>       retn
8434                                    <1>
8435                                    <1> pix_op_not_w:
8436                                    <1>       ; 31/01/2021
8437                                    <1>       ; ecx = win start pos (row, column)
8438                                    <1>       ; edx = size (rows, columns)
8439 0000ED4B E888F8FFFF               <1>       call   sysvideo_15_12 ; window preparations
8440 0000ED50 72F8                     <1>       jc     short pix_op_not_5
8441                                    <1>
8442 0000ED52 F605[88120300]20         <1>       test   byte [v_ops], 20h ; masked color 'not' ?
8443 0000ED59 7405                     <1>       jz     short pix_op_not_w_0 ; no
8444 0000ED5B E929100000               <1>       jmp    m_pix_op_not_w
8445                                    <1>                     ; window 'not' color except mask color
8446                                    <1> pix_op_not_w_0:
8447                                    <1>       ; ecx = bytes per row (to be applied)
8448                                    <1>       ; edx = screen width in bytes
8449                                    <1>       ; ebx = row count
8450                                    <1>
8451 0000ED60 8B3D[92120300]           <1>       mov    edi, [v_str]
8452                                    <1>
8453 0000ED66 803D[89120300]08         <1>       cmp    byte [v_bpp], 8 ; 8bpp
8454 0000ED6D 7707                     <1>       ja     short pix_op_not_w_1
8455                                    <1>
8456                                    <1>       ; 256 colors (8bpp)
8457 0000ED6F BD[18F30000]             <1>       mov    ebp, pix_op_not_8
8458 0000ED74 EB1E                     <1>       jmp    short pix_op_not_w_4
8459                                    <1>
8460                                    <1> pix_op_not_w_1:
8461 0000ED76 803D[89120300]18         <1>       cmp    byte [v_bpp], 24 ; 24bpp
8462 0000ED7D 7710                     <1>       ja     short pix_op_not_w_3 ; 32bpp
8463 0000ED7F 7207                     <1>       jb     short pix_op_not_w_2 ; 16bpp
8464                                    <1>
8465                                    <1>       ; 24 bit true colors
8466 0000ED81 BD[26F30000]             <1>       mov    ebp, pix_op_not_24
8467 0000ED86 EB0C                     <1>       jmp    short pix_op_not_w_4
8468                                    <1>
8469                                    <1>       ; 65536 colors (16bpp)
8470                                    <1> pix_op_not_w_2:
8471 0000ED88 BD[1EF30000]             <1>       mov    ebp, pix_op_not_16
8472 0000ED8D EB05                     <1>       jmp    short pix_op_not_w_4
8473                                    <1>
8474                                    <1>       ; 32 bit true colors
8475                                    <1> pix_op_not_w_3:
8476 0000ED8F BD[31F30000]             <1>       mov    ebp, pix_op_not_32
8477                                    <1> pix_op_not_w_4:
8478 0000ED94 E93EFFFFFF               <1>       jmp    pix_op_not_w_x
8479                                    <1>
8480                                    <1> pix_op_neg:
8481                                    <1>       ; 31/01/2021
8482                                    <1>       ; NEGATE COLOR
8483                                    <1>       ;
8484                                    <1>       ; INPUT:
8485                                    <1>       ;  ECX = start position (row, column)
8486                                    <1>       ;         (HW = row, CX = column)
8487                                    <1>       ;  EDX = size (rows, colums)
8488                                    <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>
0000ED99 F605[88120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000EDA0 7553                   <1>       jnz    short pix_op_neg_w ; window
                                <1>
0000EDA2 8B3D[8A120300]         <1>       mov    edi, [v_mem]
0000EDA8 89FE                   <1>       mov    esi, edi
0000EDAA 8B0D[8E120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000EDB0 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked negate color ?
0000EDB7 7405                   <1>       jz     short pix_op_neg_0 ; no
0000EDB9 E9FE0F0000             <1>       jmp    m_pix_op_neg ; 'neg' color except mask color
                                <1> pix_op_neg_0:
0000EDBE 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EDC5 7707                   <1>       ja     short pix_op_neg_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EDC7 E86D050000             <1>       call   pix_op_neg_8
0000EDCC EB1E                   <1>       jmp    short pix_op_neg_4
                                <1>
                                <1> pix_op_neg_1:
0000EDCE 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EDD5 7710                   <1>       ja     short pix_op_neg_3 ; 32bpp
0000EDD7 7207                   <1>       jb     short pix_op_neg_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EDD9 E869050000             <1>       call   pix_op_neg_24
0000EDDE EB0C                   <1>       jmp    short pix_op_neg_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_neg_2:
0000EDE0 E85A050000             <1>       call   pix_op_neg_16
0000EDE5 EB05                   <1>       jmp    short pix_op_neg_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_neg_3:
0000EDE7 E86D050000             <1>       call   pix_op_neg_32
                                <1> pix_op_neg_4:
0000EDEC 29F7                   <1>       sub    edi, esi
0000EDEE 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_neg_5:
0000EDF4 C3                     <1>       retn
                                <1>
                                <1> pix_op_neg_w:
                                <1>       ; 31/01/2021
                                <1>       ; ecx = win start pos (row, column)
                                <1>       ; edx = size (rows, columns)
0000EDF5 E8DEF7FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000EDFA 72F8                   <1>       jc     short pix_op_neg_5
                                <1>
0000EDFC F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked negate color ?
0000EE03 7405                   <1>       jz     short pix_op_neg_w_0 ; no
0000EE05 E937100000             <1>       jmp    m_pix_op_neg_w
                                <1>                      ; window 'neg' color except mask color
                                <1> pix_op_neg_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>
0000EE0A 8B3D[92120300]         <1>       mov    edi, [v_str]
                                <1>
0000EE10 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EE17 7707                   <1>       ja     short pix_op_neg_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EE19 BD[39F30000]           <1>       mov    ebp, pix_op_neg_8
0000EE1E EB1E                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1> pix_op_neg_w_1:
0000EE20 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EE27 7710                   <1>       ja     short pix_op_neg_w_3 ; 32bpp
0000EE29 7207                   <1>       jb     short pix_op_neg_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EE2B BD[47F30000]           <1>       mov    ebp, pix_op_neg_24
0000EE30 EB0C                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_neg_w_2:
0000EE32 BD[3FF30000]           <1>       mov    ebp, pix_op_neg_16
0000EE37 EB05                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_neg_w_3:
0000EE39 BD[59F30000]           <1>       mov    ebp, pix_op_neg_32
                                <1> pix_op_neg_w_4:
0000EE3E 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)
```

```
8594                                    <1>       ;
8595                                    <1>       ; OUTPUT:
8596                                    <1>       ;      [u.r0] will be > 0 if succesful
8597                                    <1>
8598 0000EE43 F605[88120300]10          <1>       test   byte [v_ops], 10h ; display page or window ?
8599 0000EE4A 7553                      <1>       jnz    short pix_op_inc_w ; window
8600                                    <1>
8601 0000EE4C 8B3D[8A120300]            <1>       mov    edi, [v_mem]
8602 0000EE52 89FE                      <1>       mov    esi, edi
8603 0000EE54 8B0D[8E120300]            <1>       mov    ecx, [v_siz] ; display page pixel count
8604                                    <1>
8605 0000EE5A F605[88120300]20          <1>       test   byte [v_ops], 20h ; masked increase color ?
8606 0000EE61 7405                      <1>       jz     short pix_op_inc_0 ; no
8607 0000EE63 E90C100000                <1>       jmp    m_pix_op_inc ; 'inc' color except mask color
8608                                    <1> pix_op_inc_0:
8609 0000EE68 803D[89120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
8610 0000EE6F 7707                      <1>       ja     short pix_op_inc_1
8611                                    <1>
8612                                    <1>       ; 256 colors (8bpp)
8613 0000EE71 E8EB040000                <1>       call   pix_op_inc_8
8614 0000EE76 EB1E                      <1>       jmp    short pix_op_inc_4
8615                                    <1>
8616                                    <1> pix_op_inc_1:
8617 0000EE78 803D[89120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
8618 0000EE7F 7710                      <1>       ja     short pix_op_inc_3 ; 32bpp
8619 0000EE81 7207                      <1>       jb     short pix_op_inc_2 ; 16bpp
8620                                    <1>
8621                                    <1>       ; 24 bit true colors
8622 0000EE83 E8F0040000                <1>       call   pix_op_inc_24
8623 0000EE88 EB0C                      <1>       jmp    short pix_op_inc_4
8624                                    <1>
8625                                    <1>       ; 65536 colors (16bpp)
8626                                    <1> pix_op_inc_2:
8627 0000EE8A E8DC040000                <1>       call   pix_op_inc_16
8628 0000EE8F EB05                      <1>       jmp    short pix_op_inc_4
8629                                    <1>
8630                                    <1>       ; 32 bit true colors
8631                                    <1> pix_op_inc_3:
8632 0000EE91 E8F6040000                <1>       call   pix_op_inc_32
8633                                    <1> pix_op_inc_4:
8634 0000EE96 29F7                      <1>       sub    edi, esi
8635 0000EE98 893D[64030300]            <1>       mov    [u.r0], edi
8636                                    <1> pix_op_inc_5:
8637 0000EE9E C3                        <1>       retn
8638                                    <1>
8639                                    <1> pix_op_inc_w:
8640                                    <1>       ; 31/01/2021
8641                                    <1>       ; ecx = win start pos (row, column)
8642                                    <1>       ; edx = size (rows, columns)
8643 0000EE9F E834F7FFFF                <1>       call   sysvideo_15_12 ; window preparations
8644 0000EEA4 72F8                      <1>       jc     short pix_op_inc_5
8645                                    <1>
8646 0000EEA6 F605[88120300]20          <1>       test   byte [v_ops], 20h ; masked increase color ?
8647 0000EEAD 7405                      <1>       jz     short pix_op_inc_w_0 ; no
8648 0000EEAF E959100000                <1>       jmp    m_pix_op_inc_w
8649                                    <1>                     ; window 'inc' color except mask color
8650                                    <1> pix_op_inc_w_0:
8651                                    <1>       ; ecx = bytes per row (to be applied)
8652                                    <1>       ; edx = screen width in bytes
8653                                    <1>       ; ebx = row count
8654                                    <1>
8655 0000EEB4 8B3D[92120300]            <1>       mov    edi, [v_str]
8656                                    <1>
8657 0000EEBA 803D[89120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
8658 0000EEC1 7707                      <1>       ja     short pix_op_inc_w_1
8659                                    <1>
8660                                    <1>       ; 256 colors (8bpp)
8661 0000EEC3 BD[61F30000]              <1>       mov    ebp, pix_op_inc_8
8662 0000EEC8 EB1E                      <1>       jmp    short pix_op_inc_w_4
8663                                    <1>
8664                                    <1> pix_op_inc_w_1:
8665 0000EECA 803D[89120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
8666 0000EED1 7710                      <1>       ja     short pix_op_inc_w_3 ; 32bpp
8667 0000EED3 7207                      <1>       jb     short pix_op_inc_w_2 ; 16bpp
8668                                    <1>
8669                                    <1>       ; 24 bit true colors
8670 0000EED5 BD[78F30000]              <1>       mov    ebp, pix_op_inc_24
8671 0000EEDA EB0C                      <1>       jmp    short pix_op_inc_w_4
8672                                    <1>
8673                                    <1>       ; 65536 colors (16bpp)
8674                                    <1> pix_op_inc_w_2:
8675 0000EEDC BD[6BF30000]              <1>       mov    ebp, pix_op_inc_16
8676 0000EEE1 EB05                      <1>       jmp    short pix_op_inc_w_4
8677                                    <1>
8678                                    <1>       ; 32 bit true colors
8679                                    <1> pix_op_inc_w_3:
8680 0000EEE3 BD[8CF30000]              <1>       mov    ebp, pix_op_inc_32
8681                                    <1> pix_op_inc_w_4:
8682 0000EEE8 E9EAFDFFFF                <1>       jmp    pix_op_inc_w_x
8683                                    <1>
8684                                    <1> pix_op_dec:
8685                                    <1>       ; 31/01/2021
8686                                    <1>       ; DECREASE COLOR
8687                                    <1>       ;
8688                                    <1>       ; INPUT:
8689                                    <1>       ;  ECX = start position (row, column)
8690                                    <1>       ;         (HW = row, CX = column)
8691                                    <1>       ;  EDX = size (rows, colums)
8692                                    <1>       ;         (HW = rows, DX = columns)
8693                                    <1>       ;       (0 -> invalid
8694                                    <1>       ;         (1 -> horizontal or vertical line)
8695                                    <1>       ;  [maskcolor] = mask color (to be excluded)
8696                                    <1>       ;
8697                                    <1>       ; OUTPUT:
8698                                    <1>       ;      [u.r0] will be > 0 if succesful
```

```
8699                                    <1>
8700 0000EEED F605[88120300]10          <1>       test   byte [v_ops], 10h ; display page or window ?
8701 0000EEF4 7553                      <1>       jnz    short pix_op_dec_w ; window
8702                                    <1>
8703 0000EEF6 8B3D[8A120300]            <1>       mov    edi, [v_mem]
8704 0000EEFC 89FE                      <1>       mov    esi, edi
8705 0000EEFE 8B0D[8E120300]            <1>       mov    ecx, [v_siz] ; display page pixel count
8706                                    <1>
8707 0000EF04 F605[88120300]20          <1>       test   byte [v_ops], 20h ; masked decrease color ?
8708 0000EF0B 7405                      <1>       jz     short pix_op_dec_0 ; no
8709 0000EF0D E92E100000                <1>       jmp    m_pix_op_dec ; 'dec' color except mask color
8710                                    <1> pix_op_dec_0:
8711 0000EF12 803D[89120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
8712 0000EF19 7707                      <1>       ja     short pix_op_dec_1
8713                                    <1>
8714                                    <1>       ; 256 colors (8bpp)
8715 0000EF1B E878040000                <1>       call   pix_op_dec_8
8716 0000EF20 EB1E                      <1>       jmp    short pix_op_dec_4
8717                                    <1>
8718                                    <1> pix_op_dec_1:
8719 0000EF22 803D[89120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
8720 0000EF29 7710                      <1>       ja     short pix_op_dec_3 ; 32bpp
8721 0000EF2B 7207                      <1>       jb     short pix_op_dec_2 ; 16bpp
8722                                    <1>
8723                                    <1>       ; 24 bit true colors
8724 0000EF2D E87D040000                <1>       call   pix_op_dec_24
8725 0000EF32 EB0C                      <1>       jmp    short pix_op_dec_4
8726                                    <1>
8727                                    <1>       ; 65536 colors (16bpp)
8728                                    <1> pix_op_dec_2:
8729 0000EF34 E869040000                <1>       call   pix_op_dec_16
8730 0000EF39 EB05                      <1>       jmp    short pix_op_dec_4
8731                                    <1>
8732                                    <1>       ; 32 bit true colors
8733                                    <1> pix_op_dec_3:
8734 0000EF3B E882040000                <1>       call   pix_op_dec_32
8735                                    <1> pix_op_dec_4:
8736 0000EF40 29F7                      <1>       sub    edi, esi
8737 0000EF42 893D[64030300]            <1>       mov    [u.r0], edi
8738                                    <1> pix_op_dec_5:
8739 0000EF48 C3                        <1>       retn
8740                                    <1>
8741                                    <1> pix_op_dec_w:
8742                                    <1>       ; 31/01/2021
8743                                    <1>       ; ecx = win start pos (row, column)
8744                                    <1>       ; edx = size (rows, columns)
8745 0000EF49 E88AF6FFFF                <1>       call   sysvideo_15_12 ; window preparations
8746 0000EF4E 72F8                      <1>       jc     short pix_op_dec_5
8747                                    <1>
8748 0000EF50 F605[88120300]20          <1>       test   byte [v_ops], 20h ; masked decrease color ?
8749 0000EF57 7405                      <1>       jz     short pix_op_dec_w_0 ; no
8750 0000EF59 E976100000                <1>       jmp    m_pix_op_dec_w
8751                                    <1>                      ; window 'dec' color except mask color
8752                                    <1> pix_op_dec_w_0:
8753                                    <1>       ; ecx = bytes per row (to be applied)
8754                                    <1>       ; edx = screen width in bytes
8755                                    <1>       ; ebx = row count
8756                                    <1>
8757 0000EF5E 8B3D[92120300]            <1>       mov    edi, [v_str]
8758                                    <1>
8759 0000EF64 803D[89120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
8760 0000EF6B 7707                      <1>       ja     short pix_op_dec_w_1
8761                                    <1>
8762                                    <1>       ; 256 colors (8bpp)
8763 0000EF6D BD[98F30000]              <1>       mov    ebp, pix_op_dec_8
8764 0000EF72 EB1E                      <1>       jmp    short pix_op_dec_w_4
8765                                    <1>
8766                                    <1> pix_op_dec_w_1:
8767 0000EF74 803D[89120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
8768 0000EF7B 7710                      <1>       ja     short pix_op_dec_w_3 ; 32bpp
8769 0000EF7D 7207                      <1>       jb     short pix_op_dec_w_2 ; 16bpp
8770                                    <1>
8771                                    <1>       ; 24 bit true colors
8772 0000EF7F BD[AFF30000]              <1>       mov    ebp, pix_op_dec_24
8773 0000EF84 EB0C                      <1>       jmp    short pix_op_dec_w_4
8774                                    <1>
8775                                    <1>       ; 65536 colors (16bpp)
8776                                    <1> pix_op_dec_w_2:
8777 0000EF86 BD[A2F30000]              <1>       mov    ebp, pix_op_dec_16
8778 0000EF8B EB05                      <1>       jmp    short pix_op_dec_w_4
8779                                    <1>
8780                                    <1>       ; 32 bit true colors
8781                                    <1> pix_op_dec_w_3:
8782 0000EF8D BD[C2F30000]              <1>       mov    ebp, pix_op_dec_32
8783                                    <1> pix_op_dec_w_4:
8784 0000EF92 E940FDFFFF                <1>       jmp    pix_op_dec_w_x
8785                                    <1>
8786                                    <1> pix_op_blk:
8787                                    <1>       ; 23/01/2021
8788                                    <1>       ; 22/02/2021
8789                                    <1>       ; 02/02/2021
8790                                    <1>       ; COPY PIXEL BLOCK -system to system-
8791                                    <1>       ; WRITE PIXEL BLOCKS -user to system-
8792                                    <1>       ;
8793                                    <1>       ; INPUT:
8794                                    <1>       ;   -If BL bit 5 is 0-
8795                                    <1>       ;    ECX = start position (row, column) (*)
8796                                    <1>       ;    (HW = row, CX = column)
8797                                    <1>       ;    EDX = size (rows, colums) (*)
8798                                    <1>       ;    (HW = rows, DX = columns)
8799                                    <1>       ;         (0 -> invalid)
8800                                    <1>       ;         (1 -> horizontal or vertical line)
8801                                    <1>       ;    ESI = destination (row, column) (***)
8802                                    <1>       ;   -If BL bit 5 is 1-
8803                                    <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>
0000EF97 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked or direct ?
0000EF9E 755A                   <1>       jnz    short pix_op_blk_u ; blocks from user's buffer
                                <1>
0000EFA0 89F0                   <1>       mov    eax, esi ; destination position (row, col)
0000EFA2 E8DEF6FFFF             <1>       call   calc_pixel_offset
0000EFA7 3B05[8E120300]         <1>       cmp    eax, [v_siz]
0000EFAD 734A                   <1>       jnb    short pix_op_blk_retn ; out of display page
0000EFAF 89C6                   <1>       mov    esi, eax
0000EFB1 E8ACF6FFFF             <1>       call   pixels_to_byte_count
0000EFB6 89C7                   <1>       mov    edi, eax
0000EFB8 89D0                   <1>       mov    eax, edx  ; size
0000EFBA E8C6F6FFFF             <1>       call   calc_pixel_offset
                                <1>       ; 22/02/2021
0000EFBF 3B05[8E120300]         <1>       cmp    eax, [v_siz]
0000EFC5 7732                   <1>       ja     short pix_op_blk_retn ; out of display page
0000EFC7 01C6                   <1>       add    esi, eax
0000EFC9 3B35[8E120300]         <1>       cmp    esi, [v_siz]
0000EFCF 7728                   <1>       ja     short pix_op_blk_retn ; out of display page
                                <1>
0000EFD1 033D[8A120300]         <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>
0000EFD7 E8FCF5FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000EFDC 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>
0000EFDE 8B35[92120300]         <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>
0000EFE4 29CA                   <1>       sub    edx, ecx ; screen width - window width
                                <1> pix_op_blk_0:
0000EFE6 89C8                   <1>       mov    eax, ecx
0000EFE8 0105[64030300]         <1>       add    [u.r0], eax
0000EFEE F3A4                   <1>       rep    movsb
0000EFF0 89C1                   <1>       mov    ecx, eax
0000EFF2 01D6                   <1>       add    esi, edx ; next row
0000EFF4 01D7                   <1>       add    edi, edx ; next row
0000EFF6 4B                     <1>       dec    ebx
0000EFF7 75ED                   <1>       jnz    short pix_op_blk_0
                                <1> pix_op_blk_retn:
0000EFF9 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
0000EFFA 81FA00080000           <1>       cmp    edx, 2048
0000F000 7605                   <1>       jna    short pix_op_blk_u_0
                                <1>       ; Maximum 2048 blocks
0000F002 BA00080000             <1>       mov    edx, 2048
                                <1> pix_op_blk_u_0:
0000F007 8025[88120300]DF       <1>       and    byte [v_ops], ~20h ; clear masked bit
0000F00E 890D[9A120300]         <1>       mov    [maskcolor], ecx ; save pixel color
                                <1>       ; 22/02/2021
                                <1>       ;mov   ebp, edx ; save blocks count
                                <1>       ;push  ebp
                                <1> pix_op_blk_u_next:
0000F014 52                     <1>       push   edx
0000F015 B908000000             <1>       mov    ecx, 8
0000F01A BF[A2120300]           <1>       mov    edi, buffer8 ; 8 bytes small buffer
                                <1>       ; esi = user's buffer address
0000F01F E84A2A0000             <1>       call   transfer_from_user_buffer
0000F024 72D3                   <1>       jc     short pix_op_blk_retn
0000F026 01CE                   <1>       add    esi, ecx ; 22/02/2021
0000F028 56                     <1>       push   esi
0000F029 8B15[A2120300]         <1>       mov    edx, [buffer8] ; block start pos (row,col)
0000F02F 8B35[A6120300]         <1>       mov    esi, [buffer8+4] ; block size (rows,cols)
0000F035 8B0D[9A120300]         <1>       mov    ecx, [maskcolor]
0000F03B E848FCFFFF             <1>       call   pix_op_new_w ; new (change) color (window)
0000F040 5E                     <1>       pop    esi
                                <1>       ;pop   ebp
                                <1>       ;dec   ebp
0000F041 5A                     <1>       pop    edx
0000F042 4A                     <1>       dec    edx
0000F043 75CF                   <1>       jnz    short pix_op_blk_u_next
0000F045 C3                     <1>       retn
                                <1>
                                <1> pix_op_lin:
                                <1>       ; 12/02/2021
```

```
                        <1>       ; 11/02/2021
                        <1>       ; 10/02/2021
                        <1>       ; 05/02/2021
                        <1>       ; 02/02/2021
                        <1>       ; WRITE LINE -direct-
                        <1>       ; WRITE LINE(S) -via user's buffer-
                        <1>       ;
                        <1>       ; INPUT:
                        <1>       ;   -If BL bit 5 is 0-
                        <1>       ;     CL = color (8 bit, 256 colors)
                        <1>       ;    ECX = color (16 bit and true colors)
                        <1>       ;     DX = low 12 bits - size (length)
                        <1>       ;          high 4 bits - direction or type
                        <1>       ;            0 - Horizontal line
                        <1>       ;            1 - Vertical line
                        <1>       ;          > 1 - undefined, invalid
                        <1>       ;    ESI = start position (row, column)
                        <1>       ;         (HW = row, SI = column)
                        <1>       ;   -If BL bit 5 is 1-
                        <1>       ;     CL = color (8 bit, 256 colors)
                        <1>       ;    ECX = color (16 bit and true colors)
                        <1>       ;     DX = number of lines (in user buffer)
                        <1>       ;          (limit: 2048 lines)
                        <1>       ;    ESI = user's buffer
                        <1>       ;          contains 64 bit data for lines
                        <1>       ;         START POINT: 32 bit (row, col)
                        <1>       ;         LENGTH: 32 bit
                        <1>       ;                high 16 bits - 0
                        <1>       ;                bit 0-11 - length
                        <1>       ;                bit 12-15 - type (length)
                        <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>
                        <1>       ; 10/02/2021
0000F046 F605[88120300]20 <1>     test  byte [v_ops], 20h ; masked or direct ?
0000F04D 7445             <1>     jz    short pix_op_lin_vh ; direct (v/h lines)
                        <1>
                        <1>       ; lines from user's buffer
                        <1> pix_op_lin_u:
                        <1>       ; draw lines with desired color
                        <1>       ; according to line definitions in user's buffer
0000F04F 81FA00080000   <1>       cmp   edx, 2048
0000F055 7605           <1>       jna   short pix_op_lin_u_0
                        <1>       ; Maximum 2048 lines
0000F057 BA00080000     <1>       mov   edx, 2048
                        <1> pix_op_lin_u_0:
0000F05C 890D[9A120300] <1>       mov   [maskcolor], ecx ; save pixel color
0000F062 89D5           <1>       mov   ebp, edx ; save line count
                        <1> pix_op_lin_u_next:
0000F064 B908000000     <1>       mov   ecx, 8
0000F069 BF[A2120300]   <1>       mov   edi, buffer8 ; 8 bytes small buffer
                        <1>       ; esi = user's buffer address
0000F06E E8FB290000     <1>       call  transfer_from_user_buffer
0000F073 721E           <1>       jc    short pix_op_lin_retn
0000F075 01CE           <1>       add   esi, ecx ; 11/02/2021
0000F077 56             <1>       push  esi
0000F078 8B35[A2120300] <1>       mov   esi, [buffer8] ; line start pos (row,col)
0000F07E 8B15[A6120300] <1>       mov   edx, [buffer8+4] ; line length
0000F084 8B0D[9A120300] <1>       mov   ecx, [maskcolor]
0000F08A E805000000     <1>       call  pix_op_lin_vh ; new (change) color (window)
0000F08F 5E             <1>       pop   esi
0000F090 4D             <1>       dec   ebp
0000F091 75D1           <1>       jnz   short pix_op_lin_u_next
                        <1> pix_op_lin_retn:
0000F093 C3             <1>       retn
                        <1>
                        <1> pix_op_lin_vh:
0000F094 81FA38140000   <1>       cmp   edx, 1438h ; 1920*1080 (780hx438h) limit
0000F09A 7761           <1>       ja    short pix_op_lin_err1 ; invalid type
                        <1>                             ; (for current version)
0000F09C 66F7C2FF0F     <1>       test  dx, 0FFFh
0000F0A1 745A           <1>       jz    short pix_op_lin_err1 ; zero length!
                        <1>
0000F0A3 89F0           <1>       mov   eax, esi ; start point (row, col)
0000F0A5 E8DBF5FFFF     <1>       call  calc_pixel_offset
0000F0AA 3B05[8E120300] <1>       cmp   eax, [v_siz]
0000F0B0 734B           <1>       jnb   short pix_op_lin_err1 ; out of display page!
0000F0B2 E8ABF5FFFF     <1>       call  pixels_to_byte_count
0000F0B7 89C7           <1>       mov   edi, eax ; start point offset
0000F0B9 033D[8A120300] <1>       add   edi, [v_mem] ; LFB start address
0000F0BF 89C8           <1>       mov   eax, ecx ; color
                        <1>
0000F0C1 F6C610         <1>       test  dh, 10h
0000F0C4 0F848A000000   <1>       jz    pix_op_lin_h ; Horizontal line
                        <1>
                        <1> pix_op_lin_v:
                        <1>       ; Vertical line
0000F0CA 80E60F         <1>       and   dh, 0Fh ; low 12 bits
0000F0CD 51             <1>       push  ecx ; color
0000F0CE 89D1           <1>       mov   ecx, edx
0000F0D0 0FB705[86120300] <1>     movzx  eax, word [v_width]
0000F0D7 89C3           <1>       mov   ebx, eax
                        <1>       ; 12/02/2021
0000F0D9 F7E2           <1>       mul   edx ; rows * [v_width]
0000F0DB 01F8           <1>       add   eax, edi
0000F0DD 3B05[96120300] <1>       cmp   eax, [v_end]
0000F0E3 58             <1>       pop   eax ; color
0000F0E4 7717           <1>       ja    short pix_op_lin_err1 ; out of display page
                        <1>       ; ecx = rows
0000F0E6 89CA           <1>       mov   edx, ecx
                        <1>
0000F0E8 803D[89120300]08 <1>     cmp   byte [v_bpp], 8 ; 8bpp
```

```
9014 0000F0EF 770D                <1>       ja     short pix_op_lin_v_2
9015                              <1>       ; 256 colors (1 byte per pixel)
9016 0000F0F1 010D[64030300]      <1>       add    [u.r0], ecx ; byte count
9017                              <1> pix_op_lin_v_1:
9018 0000F0F7 8807                <1>       mov    [edi], al
9019 0000F0F9 01DF                <1>       add    edi, ebx  ; next row
9020 0000F0FB E2FA                <1>       loop   pix_op_lin_v_1
9021                              <1> pix_op_lin_err1:
9022 0000F0FD C3                  <1>       retn
9023                              <1>
9024                              <1> pix_op_lin_v_2:
9025 0000F0FE 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
9026 0000F105 773A                <1>       ja     short pix_op_lin_v_6 ; 32bpp
9027 0000F107 7226                <1>       jb     short pix_op_lin_v_4 ; 16bpp
9028                              <1>
9029                              <1>       ; 24 bit true colors
9030                              <1>       ; * 3
9031 0000F109 53                  <1>       push   ebx ; screen width in pixels
9032 0000F10A D1E3                <1>       shl    ebx, 1
9033 0000F10C 011C24              <1>       add    [esp], ebx
9034 0000F10F 5B                  <1>       pop    ebx ; screen width in bytes
9035 0000F110 010D[64030300]      <1>       add    [u.r0], ecx
9036 0000F116 D1E2                <1>       shl    edx, 1
9037 0000F118 0115[64030300]      <1>       add    [u.r0], edx ; byte count
9038                              <1> pix_op_lin_v_3:
9039 0000F11E 668907              <1>       mov    [edi], ax
9040 0000F121 C1C810              <1>       ror    eax, 16
9041 0000F124 884702              <1>       mov    [edi+2], al
9042 0000F127 C1C010              <1>       rol    eax, 16
9043 0000F12A 01DF                <1>       add    edi, ebx  ; next row
9044 0000F12C E2F0                <1>       loop   pix_op_lin_v_3
9045 0000F12E C3                  <1>       retn
9046                              <1>
9047                              <1> pix_op_lin_v_4:
9048                              <1>       ; 16 bit (65536) colors
9049 0000F12F D1E3                <1>       shl    ebx, 1
9050 0000F131 D1E2                <1>       shl    edx, 1
9051 0000F133 0115[64030300]      <1>       add    [u.r0], edx
9052                              <1> pix_op_lin_v_5:
9053 0000F139 668907              <1>       mov    [edi], ax
9054 0000F13C 01DF                <1>       add    edi, ebx  ; next row
9055 0000F13E E2F9                <1>       loop   pix_op_lin_v_5
9056 0000F140 C3                  <1>       retn
9057                              <1>
9058                              <1> pix_op_lin_v_6:
9059                              <1>       ; 32 bit true colors
9060 0000F141 C1E302              <1>       shl    ebx, 2
9061 0000F144 C1E202              <1>       shl    edx, 2
9062 0000F147 0115[64030300]      <1>       add    [u.r0], edx ; byte count
9063                              <1> pix_op_lin_v_7:
9064 0000F14D 8907                <1>       mov    [edi], eax
9065 0000F14F 01DF                <1>       add    edi, ebx  ; next row
9066 0000F151 E2FA                <1>       loop   pix_op_lin_v_7
9067 0000F153 C3                  <1>       retn
9068                              <1>
9069                              <1> pix_op_lin_h:
9070                              <1>       ; Horizontal line
9071 0000F154 80E60F              <1>       and    dh, 0Fh ; low 12 bits
9072 0000F157 89D1                <1>       mov    ecx, edx
9073 0000F159 6601D6              <1>       add    si, dx ; start column + columns
9074 0000F15C 663B35[86120300]    <1>       cmp    si, [v_width] ; screen width
9075 0000F163 7711                <1>       ja     short pix_op_lin_err2 ; out of columns limit
9076                              <1>
9077 0000F165 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
9078 0000F16C 7709                <1>       ja     short pix_op_lin_h_1
9079                              <1>       ; 256 colors (1 byte per pixel)
9080 0000F16E 010D[64030300]      <1>       add    [u.r0], ecx
9081 0000F174 F3AA                <1>       rep    stosb
9082                              <1> pix_op_lin_err2:
9083 0000F176 C3                  <1>       retn
9084                              <1>
9085                              <1> pix_op_lin_h_1:
9086 0000F177 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
9087 0000F17E 7728                <1>       ja     short pix_op_lin_h_4 ; 32bpp
9088 0000F180 721A                <1>       jb     short pix_op_lin_h_3 ; 16bpp
9089                              <1>
9090                              <1>       ; 24 bit true colors
9091                              <1>       ; * 3
9092 0000F182 0115[64030300]      <1>       add    [u.r0], edx
9093 0000F188 D1E2                <1>       shl    edx, 1
9094 0000F18A 0115[64030300]      <1>       add    [u.r0], edx
9095                              <1> pix_op_lin_h_2:
9096 0000F190 66AB                <1>       stosw
9097 0000F192 C1C810              <1>       ror    eax, 16
9098 0000F195 AA                  <1>       stosb
9099 0000F196 C1C010              <1>       rol    eax, 16
9100 0000F199 E2F5                <1>       loop   pix_op_lin_h_2
9101 0000F19B C3                  <1>       retn
9102                              <1>
9103                              <1> pix_op_lin_h_3:
9104                              <1>       ; 16 bit (65536) colors
9105 0000F19C D1E2                <1>       shl    edx, 1
9106 0000F19E 0115[64030300]      <1>       add    [u.r0], edx
9107 0000F1A4 F366AB              <1>       rep    stosw
9108 0000F1A7 C3                  <1>       retn
9109                              <1>
9110                              <1> pix_op_lin_h_4:
9111                              <1>       ; 32 bit true colors
9112 0000F1A8 C1E202              <1>       shl    edx, 2
9113 0000F1AB 0115[64030300]      <1>       add    [u.r0], edx
9114 0000F1B1 F3AB                <1>       rep    stosd
9115 0000F1B3 C3                  <1>       retn
9116                              <1>
9117                              <1> pix_op_new_8:
9118                              <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>
0000F1B4 F3AA                   <1>       rep    stosb
0000F1B6 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>
0000F1B7 F366AB                 <1>       rep    stosw
0000F1BA 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>
0000F1BB 66AB                   <1>       stosw
0000F1BD C1C810                 <1>       ror    eax, 16
0000F1C0 AA                     <1>       stosb
0000F1C1 C1C010                 <1>       rol    eax, 16
0000F1C4 E2F5                   <1>       loop   pix_op_new_24
0000F1C6 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>
0000F1C7 F3AB                   <1>       rep    stosd
0000F1C9 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>
0000F1CA 88C4                   <1>       mov    ah, al
                                <1> pix_op_add_8_0:
0000F1CC 0207                   <1>       add    al, [edi]
0000F1CE 7302                   <1>       jnc    short pix_op_add_8_1
0000F1D0 B0FF                   <1>       mov    al, 0FFh ; Max. value
                                <1> pix_op_add_8_1:
0000F1D2 AA                     <1>       stosb
0000F1D3 88E0                   <1>       mov    al, ah
0000F1D5 E2F5                   <1>       loop   pix_op_add_8_0
0000F1D7 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>
0000F1D8 89C2                   <1>       mov    edx, eax
                                <1> pix_op_add_16_0:
0000F1DA 660307                 <1>       add    ax, [edi]
0000F1DD 7304                   <1>       jnc    short pix_op_add_16_1
0000F1DF 66B8FFFF               <1>       mov    ax, 0FFFFh ; Max. value
                                <1> pix_op_add_16_1:
0000F1E3 66AB                   <1>       stosw
0000F1E5 89D0                   <1>       mov    eax, edx
0000F1E7 E2F1                   <1>       loop   pix_op_add_16_0
0000F1E9 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>
0000F1EA 53                     <1>       push   ebx
0000F1EB BBFFFFFF00             <1>       mov    ebx, 0FFFFFFh
                                <1>       ;and   eax, ebx ; 0FFFFFFh
0000F1F0 89C2                   <1>       mov    edx, eax
                                <1> pix_op_add_24_0:
0000F1F2 8B07                   <1>       mov    eax, [edi]
0000F1F4 21D8                   <1>       and    eax, ebx ; 0FFFFFFh
0000F1F6 01D0                   <1>       add    eax, edx
0000F1F8 39D8                   <1>       cmp    eax, ebx
0000F1FA 7602                   <1>       jna    short pix_op_add_24_1
0000F1FC 89D8                   <1>       mov    eax, ebx ; 0FFFFFFh  ; Max. value
                                <1> pix_op_add_24_1:
0000F1FE 66AB                   <1>       stosw
0000F200 C1E810                 <1>       shr    eax, 16
0000F203 AA                     <1>       stosb
0000F204 E2EC                   <1>       loop   pix_op_add_24_0
0000F206 89D0                   <1>       mov    eax, edx
0000F208 5B                     <1>       pop    ebx
0000F209 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>
0000F20A 89C2           <1>       mov    edx, eax
                        <1> pix_op_add_32_0:
0000F20C 0307           <1>       add    eax, [edi]
0000F20E 7303           <1>       jnc    short pix_op_add_32_1
                        <1>       ;mov   eax, 0FFFFFFFFh ; Max. value
0000F210 29C0           <1>       sub    eax, eax
0000F212 48             <1>       dec    eax
                        <1> pix_op_add_32_1:
0000F213 AB             <1>       stosd
0000F214 89D0           <1>       mov    eax, edx
0000F216 E2F4           <1>       loop   pix_op_add_32_0
0000F218 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>
0000F219 88C4           <1>       mov    ah, al
                        <1> pix_op_sub_8_0:
0000F21B 8A07           <1>       mov    al, [edi]
0000F21D 28E0           <1>       sub    al, ah
0000F21F 7302           <1>       jnb    short pix_op_sub_8_1
0000F221 30C0           <1>       xor    al, al ; 0 ; Min. value
                        <1> pix_op_sub_8_1:
0000F223 AA             <1>       stosb
0000F224 E2F5           <1>       loop   pix_op_sub_8_0
0000F226 88E0           <1>       mov    al, ah
0000F228 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>
0000F229 89C2           <1>       mov    edx, eax
                        <1> pix_op_sub_16_0:
0000F22B 668B07         <1>       mov    ax, [edi]
0000F22E 6629D0         <1>       sub    ax, dx
0000F231 7302           <1>       jnb    short pix_op_sub_16_1
0000F233 31C0           <1>       xor    eax, eax ; 0 ; Min. value
                        <1> pix_op_sub_16_1:
0000F235 66AB           <1>       stosw
0000F237 E2F2           <1>       loop   pix_op_sub_16_0
0000F239 89D0           <1>       mov    eax, edx
0000F23B 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
0000F23C 89C2           <1>       mov    edx, eax
                        <1> pix_op_sub_24_0:
0000F23E 8B07           <1>       mov    eax, [edi]
                        <1>       ; 27/02/2021
0000F240 25FFFFFF00     <1>       and    eax, 0FFFFFFh
0000F245 29D0           <1>       sub    eax, edx
0000F247 7302           <1>       jnb    short pix_op_sub_24_1
0000F249 31C0           <1>       xor    eax, eax ; 0 ; Min. value
                        <1> pix_op_sub_24_1:
0000F24B 66AB           <1>       stosw
0000F24D C1E810         <1>       shr    eax, 16
0000F250 AA             <1>       stosb
0000F251 E2EB           <1>       loop   pix_op_sub_24_0
0000F253 89D0           <1>       mov    eax, edx
0000F255 C3             <1>       retn
                        <1>
                        <1> pix_op_sub_32:
                        <1>       ; 32 bit true colors
                        <1>       ; SUBTRACT PIXEL COLOR
                        <1>       ; ecx = pixel count per row
                        <1>       ; eax = color
                        <1>       ; edi = start pixel address
                        <1>
0000F256 89C2           <1>       mov    edx, eax
                        <1> pix_op_sub_32_0:
0000F258 8B07           <1>       mov    eax, [edi]
0000F25A 29D0           <1>       sub    eax, edx
0000F25C 7302           <1>       jnb    short pix_op_sub_32_1
0000F25E 31C0           <1>       xor    eax, eax ; 0 ; Min. value
                        <1> pix_op_sub_32_1:
0000F260 AB             <1>       stosd
0000F261 E2F5           <1>       loop   pix_op_sub_32_0
0000F263 89D0           <1>       mov    eax, edx
0000F265 C3             <1>       retn
                        <1>
                        <1> pix_op_or_8:
                        <1>       ; 8 bit colors (256 colors)
                        <1>       ; OR PIXEL COLOR
                        <1>       ; ecx = pixel count per row
```

```
                        <1>       ;  al = color
                        <1>       ; edi = start pixel address
                        <1>
                        <1> pix_op_or_8_0:
0000F266 0807           <1>       or    [edi], al
0000F268 47             <1>       inc   edi
0000F269 E2FB           <1>       loop  pix_op_or_8_0
0000F26B C3             <1>       retn
                        <1>
                        <1> pix_op_or_16:
                        <1>       ; 16 bit colors (65536 colors)
                        <1>       ; OR PIXEL COLOR
                        <1>       ; ecx = pixel count per row
                        <1>       ;  ax = color
                        <1>       ; edi = start pixel address
                        <1>
                        <1> pix_op_or_16_0:
0000F26C 660907         <1>       or    [edi], ax
0000F26F 47             <1>       inc   edi
0000F270 47             <1>       inc   edi
0000F271 E2F9           <1>       loop  pix_op_or_16_0
0000F273 C3             <1>       retn
                        <1>
                        <1> pix_op_or_24:
                        <1>       ; 24 bit true colors
                        <1>       ; OR PIXEL COLOR
                        <1>       ; ecx = pixel count per row
                        <1>       ; eax = color
                        <1>       ; edi = start pixel address
                        <1>
0000F274 89C2           <1>       mov   edx, eax
                        <1> pix_op_or_24_0:
0000F276 0B07           <1>       or    eax, [edi]
0000F278 66AB           <1>       stosw
0000F27A C1E810         <1>       shr   eax, 16
0000F27D AA             <1>       stosb
0000F27E 89D0           <1>       mov   eax, edx
0000F280 E2F4           <1>       loop  pix_op_or_24_0
0000F282 C3             <1>       retn
                        <1>
                        <1> pix_op_or_32:
                        <1>       ; 32 bit true colors
                        <1>       ; OR PIXEL COLOR
                        <1>       ; ecx = pixel count per row
                        <1>       ; eax = color
                        <1>       ; edi = start pixel address
                        <1>
                        <1>       ;mov  edx, eax
                        <1> pix_op_or_32_0:
                        <1>       ;or   eax, [edi]
                        <1>       ;stosd
                        <1>       ;mov  eax, edx
0000F283 0907           <1>       or    [edi], eax
0000F285 83C704         <1>       add   edi, 4
0000F288 E2F9           <1>       loop  pix_op_or_32_0
0000F28A C3             <1>       retn
                        <1>
                        <1> pix_op_and_8:
                        <1>       ; 8 bit colors (256 colors)
                        <1>       ; AND PIXEL COLOR
                        <1>       ; ecx = pixel count per row
                        <1>       ;  al = color
                        <1>       ; edi = start pixel address
                        <1>
                        <1> pix_op_and_8_0:
0000F28B 2007           <1>       and   [edi], al
0000F28D 47             <1>       inc   edi
0000F28E E2FB           <1>       loop  pix_op_and_8_0
0000F290 C3             <1>       retn
                        <1>
                        <1> pix_op_and_16:
                        <1>       ; 16 bit colors (65536 colors)
                        <1>       ; AND PIXEL COLOR
                        <1>       ; ecx = pixel count per row
                        <1>       ;  ax = color
                        <1>       ; edi = start pixel address
                        <1>
                        <1> pix_op_and_16_0:
0000F291 662107         <1>       and   [edi], ax
0000F294 47             <1>       inc   edi
0000F295 47             <1>       inc   edi
0000F296 E2F9           <1>       loop  pix_op_and_16_0
0000F298 C3             <1>       retn
                        <1>
                        <1> pix_op_and_24:
                        <1>       ; 24 bit true colors
                        <1>       ; AND PIXEL COLOR
                        <1>       ; ecx = pixel count per row
                        <1>       ; eax = color
                        <1>       ; edi = start pixel address
                        <1>
0000F299 89C2           <1>       mov   edx, eax
                        <1> pix_op_and_24_0:
0000F29B 2307           <1>       and   eax, [edi]
0000F29D 66AB           <1>       stosw
0000F29F C1E810         <1>       shr   eax, 16
0000F2A2 AA             <1>       stosb
0000F2A3 89D0           <1>       mov   eax, edx
0000F2A5 E2F4           <1>       loop  pix_op_and_24_0
0000F2A7 C3             <1>       retn
                        <1>
                        <1> pix_op_and_32:
                        <1>       ; 32 bit true colors
                        <1>       ; AND PIXEL COLOR
                        <1>       ; ecx = pixel count per row
```

```
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
                                <1>       ;mov   edx, eax
                                <1> pix_op_and_32_0:
                                <1>       ;and   eax, [edi]
                                <1>       ;stosd
                                <1>       ;mov   eax, edx
0000F2A8 2107                   <1>       and    [edi], eax
0000F2AA 83C704                 <1>       add    edi, 4
0000F2AD E2F9                   <1>       loop   pix_op_and_32_0
0000F2AF C3                     <1>       retn
                                <1>
                                <1> pix_op_xor_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; XOR PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  al = color
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_xor_8_0:
0000F2B0 3007                   <1>       xor    [edi], al
0000F2B2 47                     <1>       inc    edi
0000F2B3 E2FB                   <1>       loop   pix_op_xor_8_0
0000F2B5 C3                     <1>       retn
                                <1>
                                <1> pix_op_xor_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; XOR PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  ax = color
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_xor_16_0:
0000F2B6 663107                 <1>       xor    [edi], ax
0000F2B9 47                     <1>       inc    edi
0000F2BA 47                     <1>       inc    edi
0000F2BB E2F9                   <1>       loop   pix_op_xor_16_0
0000F2BD C3                     <1>       retn
                                <1>
                                <1> pix_op_xor_24:
                                <1>       ; 24 bit true colors
                                <1>       ; XOR PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
0000F2BE 89C2                   <1>       mov    edx, eax
                                <1> pix_op_xor_24_0:
0000F2C0 3307                   <1>       xor    eax, [edi]
0000F2C2 66AB                   <1>       stosw
0000F2C4 C1E810                 <1>       shr    eax, 16
0000F2C7 AA                     <1>       stosb
0000F2C8 89D0                   <1>       mov    eax, edx
0000F2CA E2F4                   <1>       loop   pix_op_xor_24_0
0000F2CC C3                     <1>       retn
                                <1>
                                <1> pix_op_xor_32:
                                <1>       ; 32 bit true colors
                                <1>       ; XOR PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
                                <1>       ;mov   edx, eax
                                <1> pix_op_xor_32_0:
                                <1>       ;xor   eax, [edi]
                                <1>       ;stosd
                                <1>       ;mov   eax, edx
0000F2CD 3107                   <1>       xor    [edi], eax
0000F2CF 83C704                 <1>       add    edi, 4
0000F2D2 E2F9                   <1>       loop   pix_op_xor_32_0
0000F2D4 C3                     <1>       retn
                                <1>
                                <1> pix_op_mix_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; MIX (AVERAGE) PIXEL COLORS
                                <1>       ; ecx = pixel count per row
                                <1>       ;  al = color
                                <1>       ; edi = start pixel address
                                <1>
0000F2D5 88C4                   <1>       mov    ah, al
                                <1> pix_op_mix_8_0:
0000F2D7 0207                   <1>       add    al, [edi]
0000F2D9 D0D8                   <1>       rcr    al, 1
0000F2DB AA                     <1>       stosb
0000F2DC 88E0                   <1>       mov    al, ah
0000F2DE E2F7                   <1>       loop   pix_op_mix_8_0
0000F2E0 C3                     <1>       retn
                                <1>
                                <1> pix_op_mix_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; MIX (AVERAGE) PIXEL COLORS
                                <1>       ; ecx = pixel count per row
                                <1>       ;  ax = color
                                <1>       ; edi = start pixel address
                                <1>
0000F2E1 89C2                   <1>       mov    edx, eax
                                <1> pix_op_mix_16_0:
0000F2E3 660307                 <1>       add    ax, [edi]
0000F2E6 66D1D8                 <1>       rcr    ax, 1
0000F2E9 66AB                   <1>       stosw
0000F2EB 89D0                   <1>       mov    eax, edx
0000F2ED E2F4                   <1>       loop   pix_op_mix_16_0
0000F2EF C3                     <1>       retn
```

```
                                <1>
                                <1> pix_op_mix_24:
                                <1>     ; 24 bit true colors
                                <1>     ; MIX (AVERAGE) PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = color
                                <1>     ; edi = start pixel address
                                <1>
0000F2F0 53                     <1>     push   ebx
0000F2F1 BBFFFFFF00             <1>     mov    ebx, 0FFFFFFh
                                <1>     ;and   eax, ebx ; 0FFFFFFh
0000F2F6 89C2                   <1>     mov    edx, eax
                                <1> pix_op_mix_24_0:
0000F2F8 8B07                   <1>     mov    eax, [edi]
0000F2FA 21D8                   <1>     and    eax, ebx ; 0FFFFFFh
0000F2FC 01D0                   <1>     add    eax, edx
0000F2FE D1E8                   <1>     shr    eax, 1
                                <1>     ;rcr   eax, 1
0000F300 66AB                   <1>     stosw
0000F302 C1E810                 <1>     shr    eax, 16
0000F305 AA                     <1>     stosb
0000F306 E2F0                   <1>     loop   pix_op_mix_24_0
0000F308 89D0                   <1>     mov    eax, edx
0000F30A 5B                     <1>     pop    ebx
0000F30B C3                     <1>     retn
                                <1>
                                <1> pix_op_mix_32:
                                <1>     ; 32 bit true colors
                                <1>     ; MIX (AVERAGE) PIXEL COLORS
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = color
                                <1>     ; edi = start pixel address
                                <1>
0000F30C 89C2                   <1>     mov    edx, eax
                                <1> pix_op_mix_32_0:
0000F30E 0307                   <1>     add    eax, [edi]
0000F310 D1D8                   <1>     rcr    eax, 1
0000F312 AB                     <1>     stosd
0000F313 89D0                   <1>     mov    eax, edx
0000F315 E2F7                   <1>     loop   pix_op_mix_32_0
0000F317 C3                     <1>     retn
                                <1>
                                <1> pix_op_not_8:
                                <1>     ; 8 bit colors (256 colors)
                                <1>     ; NOT PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_not_8_0:
0000F318 F617                   <1>     not    byte [edi]
0000F31A 47                     <1>     inc    edi
0000F31B E2FB                   <1>     loop   pix_op_not_8_0
0000F31D C3                     <1>     retn
                                <1>
                                <1> pix_op_not_16:
                                <1>     ; 16 bit colors (65536 colors)
                                <1>     ; NOT PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_not_16_0:
0000F31E 66F717                 <1>     not    word [edi]
0000F321 47                     <1>     inc    edi
0000F322 47                     <1>     inc    edi
0000F323 E2F9                   <1>     loop   pix_op_not_16_0
0000F325 C3                     <1>     retn
                                <1>
                                <1> pix_op_not_24:
                                <1>     ; 24 bit true colors
                                <1>     ; NOT PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_not_24_0:
0000F326 66F717                 <1>     not    word [edi]
0000F329 47                     <1>     inc    edi
0000F32A 47                     <1>     inc    edi
0000F32B F617                   <1>     not    byte [edi]
0000F32D 47                     <1>     inc    edi
0000F32E E2F6                   <1>     loop   pix_op_not_24_0
0000F330 C3                     <1>     retn
                                <1>
                                <1> pix_op_not_32:
                                <1>     ; 32 bit true colors
                                <1>     ; NOT PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; eax = color
                                <1>     ; edi = start pixel address
                                <1> pix_op_not_32_0:
0000F331 F717                   <1>     not    dword [edi]
0000F333 83C704                 <1>     add    edi, 4
0000F336 E2F9                   <1>     loop   pix_op_not_32_0
0000F338 C3                     <1>     retn
                                <1>
                                <1> pix_op_neg_8:
                                <1>     ; 8 bit colors (256 colors)
                                <1>     ; NEG PIXEL COLOR
                                <1>     ; ecx = pixel count per row
                                <1>     ; edi = start pixel address
                                <1>
                                <1> pix_op_neg_8_0:
0000F339 F61F                   <1>     neg    byte [edi]
0000F33B 47                     <1>     inc    edi
0000F33C E2FB                   <1>     loop   pix_op_neg_8_0
0000F33E C3                     <1>     retn
```

```
                                <1>
                                <1> pix_op_neg_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; NEG PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_neg_16_0:
0000F33F 66F71F                 <1>       neg   word [edi]
0000F342 47                     <1>       inc   edi
0000F343 47                     <1>       inc   edi
0000F344 E2F9                   <1>       loop  pix_op_neg_16_0
0000F346 C3                     <1>       retn
                                <1>
                                <1> pix_op_neg_24:
                                <1>       ; 24 bit true colors
                                <1>       ; NEG PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_neg_24_0:
0000F347 8B07                   <1>       mov   eax, [edi]
0000F349 25FFFFFF00             <1>       and   eax, 0FFFFFFh
0000F34E F7D8                   <1>       neg   eax
0000F350 66AB                   <1>       stosw
0000F352 C1E810                 <1>       shr   eax, 16
0000F355 AA                     <1>       stosb
0000F356 E2EF                   <1>       loop  pix_op_neg_24_0
0000F358 C3                     <1>       retn
                                <1>
                                <1> pix_op_neg_32:
                                <1>       ; 32 bit true colors
                                <1>       ; NEG PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1> pix_op_neg_32_0:
0000F359 F71F                   <1>       neg   dword [edi]
0000F35B 83C704                 <1>       add   edi, 4
0000F35E E2F9                   <1>       loop  pix_op_neg_32_0
0000F360 C3                     <1>       retn
                                <1>
                                <1> pix_op_inc_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; INCREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_inc_8_0:
0000F361 FE07                   <1>       inc   byte [edi]
0000F363 7502                   <1>       jnz   short pix_op_inc_8_1
                                <1>       ;mov  [edi], 0FFh ; Max. value
0000F365 FE0F                   <1>       dec   byte [edi]
                                <1> pix_op_inc_8_1:
0000F367 47                     <1>       inc   edi
0000F368 E2F7                   <1>       loop  pix_op_inc_8_0
0000F36A C3                     <1>       retn
                                <1>
                                <1> pix_op_inc_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; INCREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_inc_16_0:
0000F36B 66FF07                 <1>       inc   word [edi]
0000F36E 7503                   <1>       jnz   short pix_op_inc_16_1
                                <1>       ;mov  word [edi], 0FFFFh ; Max. value
0000F370 66FF0F                 <1>       dec   word [edi]
                                <1> pix_op_inc_16_1:
0000F373 47                     <1>       inc   edi
0000F374 47                     <1>       inc   edi
0000F375 E2F4                   <1>       loop  pix_op_inc_16_0
0000F377 C3                     <1>       retn
                                <1>
                                <1> pix_op_inc_24:
                                <1>       ; 24 bit true colors
                                <1>       ; INCREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_inc_24_0:
0000F378 8B07                   <1>       mov   eax, [edi]
0000F37A 40                     <1>       inc   eax
0000F37B 25FFFFFF00             <1>       and   eax, 0FFFFFFh
0000F380 7501                   <1>       jnz   short pix_op_inc_24_1
                                <1>       ;mov  eax, 0FFFFFFh  ; Max. value
0000F382 48                     <1>       dec   eax ; 0FFFFFFFFh
                                <1> pix_op_inc_24_1:
0000F383 66AB                   <1>       stosw
0000F385 C1E810                 <1>       shr   eax, 16
0000F388 AA                     <1>       stosb
0000F389 E2ED                   <1>       loop  pix_op_inc_24_0
0000F38B C3                     <1>       retn
                                <1>
                                <1> pix_op_inc_32:
                                <1>       ; 32 bit true colors
                                <1>       ; INCREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_inc_32_0:
0000F38C FF07                   <1>       inc   dword [edi]
0000F38E 7502                   <1>       jnz   short pix_op_inc_32_1
                                <1>       ;mov  dword [edi], 0FFFFFFFFh ; Max. value
```

```
9749 0000F390 FF0F              <1>       dec    dword [edi]
9750                            <1> pix_op_inc_32_1:
9751 0000F392 83C704            <1>       add    edi, 4
9752 0000F395 E2F5              <1>       loop   pix_op_inc_32_0
9753 0000F397 C3                <1>       retn
9754                            <1>
9755                            <1> pix_op_dec_8:
9756                            <1>       ; 8 bit colors (256 colors)
9757                            <1>       ; DECREASE PIXEL COLOR
9758                            <1>       ; ecx = pixel count per row
9759                            <1>       ; edi = start pixel address
9760                            <1>
9761                            <1> pix_op_dec_8_0:
9762 0000F398 FE0F              <1>       dec    byte [edi]
9763 0000F39A 7902              <1>       jns    short pix_op_dec_8_1
9764 0000F39C FE07              <1>       inc    byte [edi] ; 0 ; Min. value
9765                            <1> pix_op_dec_8_1:
9766 0000F39E 47                <1>       inc    edi
9767 0000F39F E2F7              <1>       loop   pix_op_dec_8_0
9768 0000F3A1 C3                <1>       retn
9769                            <1>
9770                            <1> pix_op_dec_16:
9771                            <1>       ; 16 bit colors (65536 colors)
9772                            <1>       ; DECREASE PIXEL COLOR
9773                            <1>       ; ecx = pixel count per row
9774                            <1>       ; edi = start pixel address
9775                            <1>
9776                            <1> pix_op_dec_16_0:
9777 0000F3A2 66FF0F            <1>       dec    word [edi]
9778 0000F3A5 7903              <1>       jns    short pix_op_dec_16_1
9779 0000F3A7 66FF07            <1>       inc    word [edi] ; 0 ; Min. value
9780                            <1> pix_op_dec_16_1:
9781 0000F3AA 47                <1>       inc edi
9782 0000F3AB 47                <1>       inc edi
9783 0000F3AC E2F4              <1>       loop   pix_op_dec_16_0
9784 0000F3AE C3                <1>       retn
9785                            <1>
9786                            <1> pix_op_dec_24:
9787                            <1>       ; 24 bit true colors
9788                            <1>       ; DECREASE PIXEL COLOR
9789                            <1>       ; ecx = pixel count per row
9790                            <1>       ; edi = start pixel address
9791                            <1>
9792                            <1> pix_op_dec_24_0:
9793 0000F3AF 8B07              <1>       mov    eax, [edi]
9794 0000F3B1 25FFFFFF00        <1>       and    eax, 0FFFFFFh
9795 0000F3B6 7401              <1>       jz     short pix_op_dec_24_1
9796                            <1>                     ; 0 ; Min. value
9797 0000F3B8 48                <1>       dec    eax
9798                            <1> pix_op_dec_24_1:
9799 0000F3B9 66AB              <1>       stosw
9800 0000F3BB C1E810            <1>       shr    eax, 16
9801 0000F3BE AA                <1>       stosb
9802 0000F3BF E2B7              <1>       loop   pix_op_inc_24_0
9803 0000F3C1 C3                <1>       retn
9804                            <1>
9805                            <1> pix_op_dec_32:
9806                            <1>       ; 32 bit true colors
9807                            <1>       ; DECREASE PIXEL COLOR
9808                            <1>       ; ecx = pixel count per row
9809                            <1>       ; edi = start pixel address
9810                            <1>
9811                            <1> pix_op_dec_32_0:
9812 0000F3C2 FF0F              <1>       dec    dword [edi]
9813 0000F3C4 7902              <1>       jns    short pix_op_dec_32_1
9814 0000F3C6 FF07              <1>       inc    dword [edi] ; 0 ; Min. value
9815                            <1> pix_op_dec_32_1:
9816 0000F3C8 83C704            <1>       add    edi, 4
9817 0000F3CB E2F5              <1>       loop   pix_op_dec_32_0
9818 0000F3CD 89D0              <1>       mov    eax, edx
9819 0000F3CF C3                <1>       retn
9820                            <1>
9821                            <1> pix_op_rpl_8:
9822                            <1>       ; 8 bit colors (256 colors)
9823                            <1>       ; REPLACE PIXEL COLORS
9824                            <1>       ; ecx = pixel count per row
9825                            <1>       ;  al = new color
9826                            <1>       ; byte [maskcolor] = old color
9827                            <1>       ; edi = start pixel address
9828                            <1>
9829 0000F3D0 8A25[9A120300]   <1>       mov    ah, [maskcolor]
9830                            <1> pix_op_rpl_8_0:
9831 0000F3D6 3A27              <1>       cmp    ah, [edi]
9832 0000F3D8 7502              <1>       jne    short pix_op_rpl_8_1
9833 0000F3DA 8807              <1>       mov    [edi], al
9834                            <1> pix_op_rpl_8_1:
9835 0000F3DC 47                <1>       inc    edi
9836 0000F3DD E2F7              <1>       loop   pix_op_rpl_8_0
9837 0000F3DF C3                <1>       retn
9838                            <1>
9839                            <1> pix_op_rpl_16:
9840                            <1>       ; 16 bit colors (65536 colors)
9841                            <1>       ; REPLACE PIXEL COLORS
9842                            <1>       ; ecx = pixel count per row
9843                            <1>       ;  ax = new color
9844                            <1>       ; word [maskcolor] = old color
9845                            <1>       ; edi = start pixel address
9846                            <1>
9847 0000F3E0 8B15[9A120300]   <1>       mov    edx, [maskcolor]
9848                            <1> pix_op_rpl_16_0:
9849 0000F3E6 663B17            <1>       cmp    dx, [edi]
9850 0000F3E9 7503              <1>       jne    short pix_op_rpl_16_1
9851 0000F3EB 668907            <1>       mov    [edi], ax
9852                            <1> pix_op_rpl_16_1:
9853 0000F3EE 47                <1>       inc    edi
```

```
9854 0000F3EF 47                <1>       inc    edi
9855 0000F3F0 E2F4              <1>       loop   pix_op_rpl_16_0
9856 0000F3F2 C3                <1>       retn
9857                            <1>
9858                            <1> pix_op_rpl_24:
9859                            <1>       ; 24 bit true colors
9860                            <1>       ; REPLACE PIXEL COLORS
9861                            <1>       ; ecx = pixel count per row
9862                            <1>       ; eax = new color
9863                            <1>       ; [maskcolor] = old color
9864                            <1>       ; edi = start pixel address
9865                            <1>
9866                            <1> pix_op_rpl_24_0:
9867 0000F3F3 8B17              <1>       mov    edx, [edi]
9868 0000F3F5 81E2FFFFFF00      <1>       and    edx, 0FFFFFFh
9869 0000F3FB 3B15[9A120300]    <1>       cmp    edx, [maskcolor]
9870 0000F401 7406              <1>       je     short pix_op_rpl_24_1
9871 0000F403 83C703            <1>       add    edi, 3
9872 0000F406 E2EB              <1>       loop   pix_op_rpl_24_0
9873 0000F408 C3                <1>       retn
9874                            <1> pix_op_rpl_24_1:
9875 0000F409 AA                <1>       stosb
9876 0000F40A C1C808            <1>       ror    eax, 8
9877 0000F40D 66AB              <1>       stosw
9878 0000F40F C1C008            <1>       rol    eax, 8
9879 0000F412 E2DF              <1>       loop   pix_op_rpl_24_0
9880 0000F414 C3                <1>       retn
9881                            <1>
9882                            <1> pix_op_rpl_32:
9883                            <1>       ; 32 bit true colors
9884                            <1>       ; REPLACE PIXEL COLORS
9885                            <1>       ; ecx = pixel count per row
9886                            <1>       ; eax = new color
9887                            <1>       ; [maskcolor] = old color
9888                            <1>       ; edi = start pixel address
9889                            <1>
9890 0000F415 8B15[9A120300]    <1>       mov    edx, [maskcolor]
9891                            <1> pix_op_rpl_32_0:
9892 0000F41B 3B17              <1>       cmp    edx, [edi]
9893 0000F41D 7504              <1>       jne    short pix_op_rpl_32_2
9894 0000F41F AB                <1>       stosd
9895 0000F420 E2F9              <1>       loop   pix_op_rpl_32_0
9896 0000F422 C3                <1>       retn
9897                            <1> pix_op_rpl_32_2:
9898 0000F423 83C704            <1>       add    edi, 4
9899 0000F426 E2F3              <1>       loop   pix_op_rpl_32_0
9900 0000F428 C3                <1>       retn
9901                            <1>
9902                            <1> pix_op_chr:
9903                            <1>       ; 15/02/2021
9904                            <1>       ; 05/02/2021
9905                            <1>       ; WRITE CHARACTER (FONT)
9906                            <1>       ; 05/01/2021 ([ufont])
9907                            <1>       ; 01/01/2021
9908                            <1>       ;     CL = char's color (8 bit, 256 colors)
9909                            <1>       ;    ECX = char's color (16 bit and true colors)
9910                            <1>       ;     DL = Character's ASCII code
9911                            <1>       ;     DH bit 0 -> font height
9912                            <1>       ;        0 -> 8x16 character font
9913                            <1>       ;        1 -> 8x8 character font
9914                            <1>       ;     DH bit 1 & 2 -> scale
9915                            <1>       ;        0 = 1/1 (8 pixels per char row)
9916                            <1>       ;        1 = 2/1 (16 pixels per char row)
9917                            <1>       ;        2 = 3/1 (24 pixels per char row)
9918                            <1>       ;        3 = 4/1 (32 pixels per char row)
9919                            <1>       ;     DH bit 6 -> [ufont] option (1 = use [ufont])
9920                            <1>       ;     If DH bit 7 = 1
9921                            <1>       ;       USER FONT (from user buffer)
9922                            <1>       ;          DL = 0 -> 8x8 (width: 1 byte per row)
9923                            <1>       ;          DL = 1 -> 8x16
9924                            <1>       ;          DL = 2 -> 16x16 (width: 2 bytes)
9925                            <1>       ;          DL = 3 -> 16x32
9926                            <1>       ;          DL = 4 -> 24x24 (width: 3 bytes)
9927                            <1>       ;          DL = 5 -> 24x48
9928                            <1>       ;          DL = 6 -> 32x32 (width: 4 bytes)
9929                            <1>       ;          DL = 7 -> 32x64
9930                            <1>       ;          DL > 7 -> invalid (unused)
9931                            <1>       ;       EDI = user's font buffer address
9932                            <1>       ;          (NOTE: byte order is as row0,row1,row2..)
9933                            <1>       ;     ESI = start position (row, column) (*)
9934                            <1>       ;           (HW = row, SI = column)
9935                            <1>
9936 0000F429 89F0              <1>       mov    eax, esi ; start position
9937 0000F42B E855F2FFFF        <1>       call   calc_pixel_offset
9938 0000F430 3B05[8E120300]    <1>       cmp    eax, [v_siz]
9939 0000F436 736D              <1>       jnb    short pix_op_chr_err ; out of display page!
9940 0000F438 E825F2FFFF        <1>       call   pixels_to_byte_count
9941                            <1>       ; eax = font start offset
9942 0000F43D 0305[8A120300]    <1>       add    eax, [v_mem] ; LFB start address
9943 0000F443 A3[92120300]      <1>       mov    [v_str], eax ; font start address
9944                            <1>
9945 0000F448 890D[9A120300]    <1>       mov    [maskcolor], ecx ; save char's color
9946                            <1>
9947 0000F44E 8835[88120300]    <1>       mov    [v_ops], dh
9948                            <1>
9949 0000F454 81E6FFFF0000      <1>       and    esi, 0FFFFh
9950 0000F45A 8935[A2120300]    <1>       mov    [buffer8], esi ; start column
9951                            <1>
9952 0000F460 31DB              <1>       xor    ebx, ebx ; 0
9953 0000F462 31C0              <1>       xor    eax, eax ; 15/02/2021
9954                            <1>
9955 0000F464 F6C680            <1>       test   dh, 80h
9956 0000F467 7577              <1>       jnz    short pix_op_chr_u ; user font
9957                            <1>
9958 0000F469 80E63F            <1>       and    dh, 3Fh ; clear bit 6, [UFONT] option bit
```

```
9959 0000F46C 7409                <1>       jz      short pix_op_chr_0
9960                              <1>
9961 0000F46E 80FE07              <1>       cmp     dh, 7
9962 0000F471 7732                <1>       ja      short pix_op_chr_err
9963                              <1>                       ; invalid (undefined) option
9964 0000F473 88F4                <1>       mov     ah, dh
9965 0000F475 D0EC                <1>       shr     ah, 1
9966                              <1>       ; ah = 0 to 3, scale
9967                              <1>       ;jmp   short pix_op_chr_font_pixels
9968                              <1>
9969                              <1> pix_op_chr_font_pixels:
9970                              <1>       ; 05/02/2021
9971                              <1>       ; write scaled font to buffer
9972                              <1>
9973                              <1>       ; DL = ASCII code of character
9974                              <1>       ; AH = scale
9975                              <1>       ; EDI = buffer address (kernel)
9976                              <1>
9977                              <1> pix_op_chr_0:
9978 0000F477 88D3                <1>       mov     bl, dl ; 15/02/2021
9979 0000F479 31C9                <1>       xor     ecx, ecx
9980 0000F47B B610                <1>       mov     dh, 16
9981 0000F47D F605[88120300]01    <1>       test    byte [v_ops], 1 ; 8x8 font ?
9982 0000F484 7428                <1>       jz      short pix_op_chr_2 ; 8x16 font
9983 0000F486 B608                <1>       mov     dh, 8
9984 0000F488 C1E303              <1>       shl     ebx, 3 ; * 8
9985 0000F48B F605[88120300]40    <1>       test    byte [v_ops], 40h ; [ufont] option
9986 0000F492 7412                <1>       jz      short pix_op_chr_1  ; no
9987                              <1>       ; test 8x8 user font is ready flag
9988 0000F494 F605[7E120300]01    <1>       test    byte [ufont], 1
9989 0000F49B 7409                <1>       jz      short pix_op_chr_1 ; no
9990 0000F49D 81C300500900        <1>       add     ebx, VGAFONT8USER
9991 0000F4A3 EB2C                <1>       jmp     short pix_op_chr_fpos_0
9992                              <1> pix_op_chr_err:
9993 0000F4A5 C3                  <1>       retn
9994                              <1> pix_op_chr_1:
9995 0000F4A6 81C3[845F0100]      <1>       add     ebx, vgafont8 ; system font (8x8)
9996 0000F4AC EB23                <1>       jmp     short pix_op_chr_fpos_0
9997                              <1> pix_op_chr_2:
9998 0000F4AE C1E304              <1>       shl     ebx, 4 ; * 16
9999 0000F4B1 F605[88120300]40    <1>       test    byte [v_ops], 40h ; [ufont] option
10000 0000F4B8 7411               <1>       jz      short pix_op_chr_3  ; no
10001                             <1>       ; test 8x16 user font is ready flag
10002 0000F4BA F605[7E120300]02   <1>       test    byte [ufont], 2
10003 0000F4C1 7408               <1>       jz      short pix_op_chr_3 ; no
10004 0000F4C3 81C300400900       <1>       add     ebx, VGAFONT16USER
10005 0000F4C9 EB06               <1>       jmp     short pix_op_chr_fpos_0
10006                             <1> pix_op_chr_3:
10007 0000F4CB 81C3[84750100]     <1>       add     ebx, vgafont16 ; system font (8x16)
10008                             <1> pix_op_chr_fpos_0:
10009 0000F4D1 20E4               <1>       and     ah, ah
10010 0000F4D3 754B               <1>       jnz     short pix_op_chr_fpos_1 ; scale > 1
10011                             <1>       ; no scale (scale = 1)
10012 0000F4D5 89DE               <1>       mov     esi, ebx ; 15/02/2021
10013 0000F4D7 88F1               <1>       mov     cl, dh ; rows/height (16 or 8)
10014 0000F4D9 B608               <1>       mov     dh, 8  ; columns/width
10015 0000F4DB E9CD000000         <1>       jmp     pix_op_chr_f2p
10016                             <1> pix_op_chr_u:
10017                             <1>       ; write user defined font
10018 0000F4E0 80FE80             <1>       cmp     dh, 80h
10019 0000F4E3 75C0               <1>       jne     short pix_op_chr_err
10020 0000F4E5 80FA07             <1>       cmp     dl, 7
10021 0000F4E8 77BB               <1>       ja      short pix_op_chr_err
10022                             <1>
10023                             <1>       ; 16/02/2021
10024 0000F4EA 89FE               <1>       mov     esi, edi ; user's font buffer
10025                             <1>
10026                             <1>       ;xor   eax, eax
10027                             <1>       ; eax = 0 ; 15/02/2021
10028 0000F4EC 88D4               <1>       mov     ah, dl
10029 0000F4EE D0EC               <1>       shr     ah, 1
10030 0000F4F0 FEC4               <1>       inc     ah
10031                             <1>       ; ah =  1 to 4
10032 0000F4F2 88E0               <1>       mov     al, ah
10033 0000F4F4 C0E003             <1>       shl     al, 3 ; * 8
10034                             <1>       ; al = 8,16,24,32
10035 0000F4F7 88C3               <1>       mov     bl, al
10036 0000F4F9 88C7               <1>       mov     bh, al
10037 0000F4FB F6E4               <1>       mul     ah
10038                             <1>       ; ax = 8,32,72,128 bytes
10039 0000F4FD F6C201             <1>       test    dl, 1
10040 0000F500 7405               <1>       jz      short pix_op_chr_u_0
10041 0000F502 66D1E0             <1>       shl     ax, 1 ; *2
10042                             <1>       ; ax = 16,32,144,256 bytes
10043 0000F505 D0E7               <1>       shl     bh, 1
10044                             <1> pix_op_chr_u_0:
10045                             <1>       ; eax = byte count
10046 0000F507 89C1               <1>       mov     ecx, eax
10047 0000F509 BF00760900         <1>       mov     edi, VBE3SAVERESTOREBLOCK
10048                             <1>       ; esi = user buffer
10049 0000F50E E85B250000         <1>       call    transfer_from_user_buffer
10050 0000F513 7290               <1>       jc      short pix_op_chr_err
10051                             <1>
10052 0000F515 88F9               <1>       mov     cl, bh ; rows/height
10053 0000F517 88DE               <1>       mov     dh, bl ; columns (width)
10054 0000F519 89FE               <1>       mov     esi, edi ; VBE3SAVERESTOREBLOCK
10055 0000F51B E98D000000         <1>       jmp     pix_op_chr_f2p
10056                             <1>
10057                             <1> pix_op_chr_fpos_1:
10058                             <1>       ; 18/02/2021
10059                             <1>       ; scale > 1
10060 0000F520 88F5               <1>       mov     ch, dh ; 16 or 8
10061 0000F522 BF00760900         <1>       mov     edi, VBE3SAVERESTOREBLOCK
10062 0000F527 89FE               <1>       mov     esi, edi
10063 0000F529 FECC               <1>       dec     ah
```

```
10064 0000F52B 7523              <1>       jnz    short pix_op_chr_fpos_5 ; scale > 2
10065                            <1>       ; scale = 2
10066                            <1> pix_op_chr_fpos_2:
10067 0000F52D B108              <1>       mov    cl, 8
10068 0000F52F 8A13              <1>       mov    dl, [ebx]
10069                            <1> pix_op_chr_fpos_3:
10070 0000F531 66C1E002          <1>       shl    ax, 2
10071 0000F535 D0E2              <1>       shl    dl, 1
10072 0000F537 7302              <1>       jnc    short pix_op_chr_fpos_4
10073 0000F539 0C03              <1>       or     al, 3
10074                            <1> pix_op_chr_fpos_4:
10075 0000F53B FEC9              <1>       dec    cl
10076 0000F53D 75F2              <1>       jnz    short pix_op_chr_fpos_3
10077 0000F53F 66AB              <1>       stosw
10078                            <1>       ; 18/02/2021
10079 0000F541 66AB              <1>       stosw
10080 0000F543 43                <1>       inc    ebx
10081 0000F544 FECD              <1>       dec    ch
10082 0000F546 75E5              <1>       jnz    short pix_op_chr_fpos_2
10083                            <1>       ; scale = 2
10084 0000F548 88F1              <1>       mov    cl, dh ; 16 or 8 (height/rows)
10085 0000F54A D0E1              <1>       shl    cl, 1  ; 32 or 16 rows
10086 0000F54C B610              <1>       mov    dh, 16 ; columns (width)
10087 0000F54E EB5D              <1>       jmp    short pix_op_chr_f2p
10088                            <1> pix_op_chr_fpos_5:
10089 0000F550 FECC              <1>       dec    ah
10090 0000F552 7538              <1>       jnz    short pix_op_chr_fpos_9 ; scale = 4
10091                            <1>       ; scale = 3
10092                            <1> pix_op_chr_fpos_6:
10093 0000F554 B108              <1>       mov    cl, 8
10094 0000F556 8A13              <1>       mov    dl, [ebx]
10095                            <1> pix_op_chr_fpos_7:
10096 0000F558 C1E003            <1>       shl    eax, 3
10097 0000F55B D0E2              <1>       shl    dl, 1 ; 18/02/2021
10098 0000F55D 7302              <1>       jnc    short pix_op_chr_fpos_8
10099 0000F55F 0C07              <1>       or     al, 7
10100                            <1> pix_op_chr_fpos_8:
10101 0000F561 FEC9              <1>       dec    cl
10102 0000F563 75F3              <1>       jnz    short pix_op_chr_fpos_7
10103 0000F565 66AB              <1>       stosw
10104                            <1>       ; 18/02/2021
10105 0000F567 C1C810            <1>       ror    eax, 16
10106 0000F56A AA                <1>       stosb
10107 0000F56B C1C010            <1>       rol    eax, 16
10108 0000F56E 66AB              <1>       stosw
10109 0000F570 C1C810            <1>       ror    eax, 16
10110 0000F573 AA                <1>       stosb
10111 0000F574 C1C010            <1>       rol    eax, 16
10112 0000F577 66AB              <1>       stosw
10113 0000F579 C1E810            <1>       shr    eax, 16 ; 27/02/2021
10114 0000F57C AA                <1>       stosb
10115 0000F57D 43                <1>       inc    ebx
10116                            <1>       ; 18/02/2021
10117 0000F57E FECD              <1>       dec    ch
10118 0000F580 75D2              <1>       jnz    short pix_op_chr_fpos_6
10119                            <1>       ; scale = 3
10120 0000F582 88F1              <1>       mov    cl, dh ; 16 or 8 (height/rows)
10121 0000F584 D0E1              <1>       shl    cl, 1
10122 0000F586 00F1              <1>       add    cl, dh ; 48 or 24 rows
10123 0000F588 B618              <1>       mov    dh, 24 ; columns (width)
10124 0000F58A EB21              <1>       jmp    short pix_op_chr_f2p
10125                            <1>
10126                            <1> pix_op_chr_fpos_9:
10127                            <1>       ; scale = 4
10128 0000F58C B108              <1>       mov    cl, 8
10129 0000F58E 8A13              <1>       mov    dl, [ebx]
10130                            <1> pix_op_chr_fpos_10:
10131                            <1>       ; 18/02/2021
10132 0000F590 C1E004            <1>       shl    eax, 4
10133 0000F593 D0E2              <1>       shl    dl, 1 ; 18/02/2021
10134 0000F595 7302              <1>       jnc    short pix_op_chr_fpos_11
10135 0000F597 0C0F              <1>       or     al, 0Fh ; or al, 15
10136                            <1> pix_op_chr_fpos_11:
10137 0000F599 FEC9              <1>       dec    cl
10138 0000F59B 75F3              <1>       jnz    short pix_op_chr_fpos_10
10139 0000F59D AB                <1>       stosd
10140                            <1>       ; 18/02/2021
10141 0000F59E AB                <1>       stosd
10142 0000F59F AB                <1>       stosd
10143 0000F5A0 AB                <1>       stosd
10144 0000F5A1 43                <1>       inc    ebx
10145 0000F5A2 FECD              <1>       dec    ch
10146 0000F5A4 75E6              <1>       jnz    short pix_op_chr_fpos_9
10147                            <1>       ; scale = 4
10148 0000F5A6 88F1              <1>       mov    cl, dh ; 16 or 8 (height/rows)
10149 0000F5A8 C0E102            <1>       shl    cl, 2 ; 64 or 32 rows
10150 0000F5AB B620              <1>       mov    dh, 32 ; columns (width)
10151                            <1>       ;jmp   short pix_op_chr_f2p
10152                            <1>
10153                            <1> pix_op_chr_f2p:
10154                            <1>       ; write font pixels
10155 0000F5AD 8B3D[92120300]    <1>       mov    edi, [v_str]
10156                            <1>       ; 15/02/2021
10157                            <1> pix_op_chr_f2p_next:
10158 0000F5B3 80FE08            <1>       cmp    dh, 8
10159 0000F5B6 7706              <1>       ja     short pix_op_chr_f2p_24
10160                            <1> pix_op_chr_f2p_8:
10161 0000F5B8 AC                <1>       lodsb
10162 0000F5B9 C1E018            <1>       shl    eax, 24 ; 15/02/2021
10163 0000F5BC EB16              <1>       jmp    short pix_op_chr_f2p_0
10164                            <1> pix_op_chr_f2p_24:
10165 0000F5BE 80FE18            <1>       cmp    dh, 24
10166 0000F5C1 7710              <1>       ja     short pix_op_chr_f2p_32
10167 0000F5C3 7207              <1>       jb     short pix_op_chr_f2p_16
10168                            <1>       ; 27/02/2021
```

```
10169                                  <1>       ;mov   eax, [esi]
10170 0000F5C5 AD                      <1>       lodsd
10171 0000F5C6 C1E008                  <1>       shl    eax, 8
10172                                  <1>       ;add   esi, 3
10173 0000F5C9 4E                      <1>       dec    esi
10174 0000F5CA EB08                    <1>       jmp    short pix_op_chr_f2p_0
10175                                  <1> pix_op_chr_f2p_16:
10176 0000F5CC 66AD                    <1>       lodsw
10177 0000F5CE C1E010                  <1>       shl    eax, 16 ; 15/02/2021
10178 0000F5D1 EB01                    <1>       jmp    short pix_op_chr_f2p_0
10179                                  <1> pix_op_chr_f2p_32:
10180 0000F5D3 AD                      <1>       lodsd
10181                                  <1> pix_op_chr_f2p_0:
10182                                  <1>       ; EAX = font row (8,16,24,32 pixels)
10183                                  <1>       ;      (bits are shifted to left)
10184                                  <1>       ; CL = rows
10185                                  <1>       ; DH = bits per row (8,16,24,32)
10186 0000F5D4 8B1D[A2120300]          <1>       mov    ebx, [buffer8] ; start column
10187 0000F5DA 57                      <1>       push   edi ; *
10188 0000F5DB 52                      <1>       push   edx ; **
10189                                  <1> pix_op_chr_f2p_1:
10190 0000F5DC E82E000000              <1>       call   pix_op_chr_w_pixel
10191                                  <1> pix_op_chr_f2p_2:
10192 0000F5E1 663B1D[86120300]        <1>       cmp    bx, [v_width] ; current column
10193 0000F5E8 7304                    <1>       jnb    short pix_op_chr_f2p_3
10194 0000F5EA FECE                    <1>       dec    dh
10195 0000F5EC 75EE                    <1>       jnz    short pix_op_chr_f2p_1 ; next bit
10196                                  <1> pix_op_chr_f2p_3:
10197                                  <1>       ;mov   ebx, [buffer8]
10198 0000F5EE 5A                      <1>       pop    edx ; **
10199 0000F5EF 58                      <1>       pop    eax ; *
10200 0000F5F0 3B3D[96120300]          <1>       cmp    edi, [v_end]
10201 0000F5F6 7316                    <1>       jnb    short pix_op_chr_f2p_4
10202 0000F5F8 FEC9                    <1>       dec    cl
10203 0000F5FA 7412                    <1>       jz     short pix_op_chr_f2p_4
10204                                  <1>       ; 27/02/2021
10205 0000F5FC 89C7                    <1>       mov    edi, eax
10206 0000F5FE 0FB705[86120300]        <1>       movzx  eax, word [v_width]
10207 0000F605 E858F0FFFF              <1>       call   pixels_to_byte_count
10208 0000F60A 01C7                    <1>       add    edi, eax ; next position
10209 0000F60C EBA5                    <1>       jmp    short pix_op_chr_f2p_next
10210                                  <1> pix_op_chr_f2p_4:
10211 0000F60E C3                      <1>       retn
10212                                  <1>
10213                                  <1> pix_op_chr_w_pixel:
10214                                  <1>       ; 15/02/2021
10215 0000F60F 89C5                    <1>       mov    ebp, eax
10216 0000F611 A1[9A120300]            <1>       mov    eax, [maskcolor]
10217 0000F616 803D[89120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
10218 0000F61D 7711                    <1>       ja     short pix_op_chr_wp_2
10219                                  <1>       ; 256 colors (1 byte per pixel)
10220 0000F61F D1E5                    <1>       shl    ebp, 1
10221 0000F621 7302                    <1>       jnc    short pix_op_chr_wp_0
10222 0000F623 8807                    <1>       mov    [edi], al
10223                                  <1> pix_op_chr_wp_0:
10224 0000F625 47                      <1>       inc    edi
10225 0000F626 FF05[64030300]          <1>       inc    dword [u.r0] ; +1
10226                                  <1> pix_op_chr_wp_1:
10227 0000F62C 43                      <1>       inc    ebx
10228 0000F62D 89E8                    <1>       mov    eax, ebp
10229 0000F62F C3                      <1>       retn
10230                                  <1> pix_op_chr_wp_2:
10231 0000F630 803D[89120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
10232 0000F637 772E                    <1>       ja     short pix_op_chr_wp_6 ; 32bpp
10233 0000F639 721C                    <1>       jb     short pix_op_chr_wp_4 ; 16bpp
10234                                  <1>       ; 24 bit true colors
10235                                  <1>       ; * 3
10236 0000F63B D1E5                    <1>       shl    ebp, 1
10237 0000F63D 7309                    <1>       jnc    short pix_op_chr_wp_3
10238 0000F63F 668907                  <1>       mov    [edi], ax
10239 0000F642 C1E810                  <1>       shr    eax, 16
10240 0000F645 884702                  <1>       mov    [edi+2], al
10241                                  <1> pix_op_chr_wp_3:
10242 0000F648 B803000000              <1>       mov    eax, 3 ; 27/02/2021
10243 0000F64D 01C7                    <1>       add    edi, eax  ; add edi, 3
10244 0000F64F 0105[64030300]          <1>       add    [u.r0], eax ; +3
10245                                  <1>
10246 0000F655 EBD5                    <1>       jmp    short pix_op_chr_wp_1
10247                                  <1>
10248                                  <1> pix_op_chr_wp_4:
10249                                  <1>       ; 16 bit (65536) colors
10250 0000F657 D1E5                    <1>       shl    ebp, 1
10251 0000F659 7303                    <1>       jnc    short pix_op_chr_wp_5
10252 0000F65B 668907                  <1>       mov    [edi], ax
10253                                  <1> pix_op_chr_wp_5:
10254 0000F65E 47                      <1>       inc    edi
10255 0000F65F FF05[64030300]          <1>       inc    dword [u.r0] ; +1
10256 0000F665 EBBE                    <1>       jmp    short pix_op_chr_wp_0
10257                                  <1>
10258                                  <1> pix_op_chr_wp_6:
10259                                  <1>       ; 32 bit true colors
10260 0000F667 D1E5                    <1>       shl    ebp, 1
10261 0000F669 7302                    <1>       jnc    short pix_op_chr_wp_7
10262 0000F66B 8907                    <1>       mov    [edi], eax
10263                                  <1> pix_op_chr_wp_7:
10264 0000F66D 31C0                    <1>       xor    eax, eax
10265 0000F66F B004                    <1>       mov    al, 4
10266 0000F671 01C7                    <1>       add    edi, eax ; add edi, 4
10267 0000F673 0105[64030300]          <1>       add    [u.r0], eax ; +4
10268 0000F679 EBB1                    <1>       jmp    short pix_op_chr_wp_1
10269                                  <1>
10270                                  <1> m_pix_op_cpy:
10271                                  <1>       ; 26/02/2021
10272                                  <1>       ; 06/02/2021
10273                                  <1>       ; MASKED COPY PIXELS (full screen)
```

```
10274                                  <1>       ;
10275                                  <1>       ; jump from pix_op_cpy
10276                                  <1>       ;
10277                                  <1>       ; INPUT:
10278                                  <1>       ;   ecx = transfer count (bytes)
10279                                  <1>       ;   edi = [v_mem] = start address of LFB
10280                                  <1>       ;   esi = user's buffer address (virtual)
10281                                  <1>       ;
10282                                  <1>       ; OUTPUT:
10283                                  <1>       ;   [u.r0] will be > 0 if succesful
10284                                  <1>
10285                                  <1>       ; Full screen masked copy
10286                                  <1>
10287                                  <1>       ; Modified regs: eax, ebx, edx, esi, edi, ecx
10288                                  <1>
10289                                  <1> m_pix_op_cpy_0:
10290                                  <1>       ;push  ebx ; *** ; 26/02/2021
10291 0000F67B 57                      <1>       push   edi ; **
10292 0000F67C 51                      <1>       push   ecx ; *
10293 0000F67D 81F9F8070000            <1>       cmp    ecx, 2040 ; (3*680) ; 26/02/2021
10294 0000F683 7605                    <1>       jna    short m_pix_op_cpy_1
10295 0000F685 B9F8070000              <1>       mov    ecx, 2040
10296                                  <1> m_pix_op_cpy_1:
10297 0000F68A BF00760900              <1>       mov    edi, VBE3SAVERESTOREBLOCK ; temporary buff
10298 0000F68F E8DA230000              <1>       call   transfer_from_user_buffer
10299 0000F694 726C                    <1>       jc     short m_pix_op_cpy_3
10300 0000F696 01CE                    <1>       add    esi, ecx
10301 0000F698 89F5                    <1>       mov    ebp, esi  ; save user's buffer address
10302 0000F69A 89FE                    <1>       mov    esi, edi
10303 0000F69C 89CB                    <1>       mov    ebx, ecx
10304 0000F69E 59                      <1>       pop    ecx ; *
10305 0000F69F 29D9                    <1>       sub    ecx, ebx
10306 0000F6A1 5F                      <1>       pop    edi ; **
10307 0000F6A2 31D2                    <1>       xor    edx, edx ; 26/02/2021
10308 0000F6A4 803D[89120300]08        <1>       cmp    byte [v_bpp], 8
10309 0000F6AB 7435                    <1>       je     short m_pix_op_cpy_1_8
10310 0000F6AD 803D[89120300]18        <1>       cmp    byte [v_bpp], 24
10311 0000F6B4 776D                    <1>       ja     short m_pix_op_cpy_1_32
10312 0000F6B6 724D                    <1>       jb     short m_pix_op_cpy_1_16
10313                                  <1> m_pix_op_cpy_1_24:
10314                                  <1>       ; 24 bit masked copy
10315                                  <1>       ;mov   edx, 3
10316 0000F6B8 B203                    <1>       mov    dl, 3 ; 26/02/2021
10317                                  <1> m_pix_op_cpy_1_24_0:
10318 0000F6BA 66AD                    <1>       lodsw
10319 0000F6BC C1E010                  <1>       shl    eax, 16
10320 0000F6BF AC                      <1>       lodsb
10321 0000F6C0 C1C010                  <1>       rol    eax, 16
10322 0000F6C3 3B05[9A120300]          <1>       cmp    eax, [maskcolor]
10323 0000F6C9 740F                    <1>       je     short m_pix_op_cpy_1_24_1 ; exclude
10324 0000F6CB 668907                  <1>       mov    [edi], ax
10325 0000F6CE C1E810                  <1>       shr    eax, 16
10326 0000F6D1 884702                  <1>       mov    [edi+2], al
10327 0000F6D4 0115[64030300]          <1>       add    [u.r0], edx ; +3
10328                                  <1> m_pix_op_cpy_1_24_1:
10329 0000F6DA 01D7                    <1>       add    edi, edx ; +3
10330 0000F6DC 29D3                    <1>       sub    ebx, edx ; sub ebx, 3
10331 0000F6DE 77DA                    <1>       ja     short m_pix_op_cpy_1_24_0
10332 0000F6E0 EB15                    <1>       jmp    short m_pix_op_cpy_2
10333                                  <1>
10334                                  <1> m_pix_op_cpy_1_8:
10335                                  <1>       ; 8 bit masked copy
10336 0000F6E2 AC                      <1>       lodsb
10337 0000F6E3 3A05[9A120300]          <1>       cmp    al, [maskcolor]
10338 0000F6E9 7408                    <1>       je     short m_pix_op_cpy_1_8_1 ; exclude
10339 0000F6EB 8807                    <1>       mov    [edi], al
10340 0000F6ED FF05[64030300]          <1>       inc    dword [u.r0] ; +1
10341                                  <1> m_pix_op_cpy_1_8_1:
10342 0000F6F3 47                      <1>       inc    edi ; +1
10343 0000F6F4 4B                      <1>       dec    ebx
10344 0000F6F5 75EB                    <1>       jnz    short m_pix_op_cpy_1_8
10345                                  <1> m_pix_op_cpy_2:
10346 0000F6F7 89EE                    <1>       mov    esi, ebp ; restore user's buffer addr
10347 0000F6F9 09C9                    <1>       or     ecx, ecx
10348                                  <1>       ; 26/02/2021
10349 0000F6FB 7407                    <1>       jz     short m_pix_of_cpy_4
10350 0000F6FD E979FFFFFF              <1>       jmp    m_pix_op_cpy_0
10351                                  <1> m_pix_op_cpy_3:
10352 0000F702 59                      <1>       pop    ecx ; *
10353 0000F703 5F                      <1>       pop    edi ; **
10354                                  <1> m_pix_of_cpy_4:
10355                                  <1>       ;pop   ebx ; *** ; 26/02/2021
10356 0000F704 C3                      <1>       retn
10357                                  <1>
10358                                  <1> m_pix_op_cpy_1_16:
10359                                  <1>       ; 16 bit masked copy
10360                                  <1>       ;mov   edx, 2
10361 0000F705 B202                    <1>       mov    dl, 2 ; 26/02/2021
10362                                  <1> m_pix_op_cpy_1_16_0:
10363 0000F707 66AD                    <1>       lodsw
10364 0000F709 663B05[9A120300]        <1>       cmp    ax, [maskcolor]
10365 0000F710 7409                    <1>       je     short m_pix_op_cpy_1_16_1 ; exclude
10366 0000F712 668907                  <1>       mov    [edi], ax
10367 0000F715 0115[64030300]          <1>       add    [u.r0], edx ; +2
10368                                  <1> m_pix_op_cpy_1_16_1:
10369 0000F71B 01D7                    <1>       add    edi, edx ; +2
10370 0000F71D 29D3                    <1>       sub    ebx, edx ; sub ebx, 2
10371 0000F71F 77E6                    <1>       ja     short m_pix_op_cpy_1_16_0
10372 0000F721 EBD4                    <1>       jmp    short m_pix_op_cpy_2
10373                                  <1>
10374                                  <1> m_pix_op_cpy_1_32:
10375                                  <1>       ; 32 bit masked copy
10376                                  <1>       ;mov   edx, 4
10377 0000F723 B204                    <1>       mov    dl, 4 ; 26/02/2021
10378                                  <1> m_pix_op_cpy_1_32_0:
```

```
10379 0000F725 AD                  <1>       lodsd
10380 0000F726 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
10381 0000F72C 7408                <1>       je     short m_pix_op_cpy_1_32_1 ; exclude
10382 0000F72E 8907                <1>       mov    [edi], eax
10383 0000F730 0115[64030300]      <1>       add    [u.r0], edx ; +4
10384                              <1> m_pix_op_cpy_1_32_1:
10385 0000F736 01D7                <1>       add    edi, edx ; +4
10386 0000F738 29D3                <1>       sub    ebx, edx ; sub ebx, 4
10387 0000F73A 77E9                <1>       ja     short m_pix_op_cpy_1_32_0
10388 0000F73C EBB9                <1>       jmp    short m_pix_op_cpy_2
10389                              <1>
10390                              <1> m_pix_op_cpy_w:
10391                              <1>       ; 26/02/2021
10392                              <1>       ; 06/02/2021
10393                              <1>       ; MASKED COPY PIXELS (window)
10394                              <1>       ;
10395                              <1>       ; jump from pix_op_cpy_w
10396                              <1>       ;
10397                              <1>       ; INPUT:
10398                              <1>       ;   ecx = bytes per row (to be applied)
10399                              <1>       ;   edx = screen width in bytes
10400                              <1>       ;   ebx = row count
10401                              <1>       ;   esi = user's buffer address
10402                              <1>       ;   [v_str] = window start address
10403                              <1>       ;
10404                              <1>       ; OUTPUT:
10405                              <1>       ;   [u.r0] will be > 0 if succesful
10406                              <1>
10407                              <1>       ; Window masked copy
10408                              <1>
10409                              <1> m_pix_op_cpy_w_0:
10410 0000F73E 8B3D[92120300]      <1>       mov    edi, [v_str]
10411                              <1> m_pix_op_cpy_w_1:
10412 0000F744 57                  <1>       push   edi
10413 0000F745 56                  <1>       push   esi
10414 0000F746 53                  <1>       push   ebx
10415 0000F747 52                  <1>       push   edx
10416 0000F748 51                  <1>       push   ecx
10417 0000F749 E82DFFFFFF          <1>       call   m_pix_op_cpy ; 26/02/2021
10418 0000F74E 59                  <1>       pop    ecx
10419 0000F74F 5A                  <1>       pop    edx
10420 0000F750 5B                  <1>       pop    ebx
10421 0000F751 5E                  <1>       pop    esi
10422 0000F752 5F                  <1>       pop    edi
10423 0000F753 7209                <1>       jc     short m_pix_op_cpy_w_2
10424 0000F755 4B                  <1>       dec    ebx
10425 0000F756 7406                <1>       jz     short m_pix_op_cpy_w_2 ; ok.
10426                              <1>       ; next row
10427 0000F758 01CE                <1>       add    esi, ecx ; next row in user's buffer
10428 0000F75A 01D7                <1>       add    edi, edx ; next row of window (system)
10429 0000F75C EBE6                <1>       jmp    short m_pix_op_cpy_w_1
10430                              <1> m_pix_op_cpy_w_2:
10431 0000F75E C3                  <1>       retn
10432                              <1>
10433                              <1> m_pix_op_new:
10434                              <1>       ; 06/02/2021
10435                              <1>       ; CHANGE COLOR (MASKED, full screen)
10436                              <1>       ;
10437                              <1>       ; jump from pix_op_new
10438                              <1>       ;
10439                              <1>       ; INPUT:
10440                              <1>       ;   eax = color (AL, AX, EAX)
10441                              <1>       ;   ecx = [v_siz] ; display page pixel count
10442                              <1>       ;   esi = edi = [v_mem] ; LFB start address
10443                              <1>       ;
10444                              <1>       ;   [maskcolor] = mask color (to be excluded)
10445                              <1>       ;
10446                              <1>       ; OUTPUT:
10447                              <1>       ;     [u.r0] will be > 0 if succesful
10448                              <1>
10449                              <1>       ; Full screen
10450                              <1> m_pix_op_new_0:
10451 0000F75F 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
10452 0000F766 7717                <1>       ja     short m_pix_op_new_1
10453                              <1>       ; 256 colors (8bpp)
10454                              <1>       ;jmp   short m_pix_op_new_8
10455                              <1> m_pix_op_new_8:
10456                              <1>       ; 8 bit colors (256 colors)
10457 0000F768 88C2                <1>       mov    dl, al ; new color
10458                              <1> m_pix_op_new_8_0:
10459 0000F76A AC                  <1>       lodsb
10460 0000F76B 3A05[9A120300]      <1>       cmp    al, [maskcolor]
10461 0000F771 7408                <1>       je     short m_pix_op_new_8_1 ; exclude
10462 0000F773 8817                <1>       mov    [edi], dl
10463 0000F775 FF05[64030300]      <1>       inc    dword [u.r0]
10464                              <1> m_pix_op_new_8_1:
10465 0000F77B 47                  <1>       inc    edi
10466 0000F77C E2EC                <1>       loop   m_pix_op_new_8_0
10467 0000F77E C3                  <1>       retn
10468                              <1> m_pix_op_new_1:
10469 0000F77F 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
10470 0000F786 774B                <1>       ja     short m_pix_op_new_3 ; 32bpp
10471 0000F788 722C                <1>       jb     short m_pix_op_new_2 ; 16bpp
10472                              <1>       ; 24 bit true colors
10473                              <1>       ;jmp   short m_pix_op_new_24
10474                              <1> m_pix_op_new_24:
10475                              <1>       ; 24 bit true colors
10476 0000F78A 89C2                <1>       mov    edx, eax ; new color
10477                              <1> m_pix_op_new_24_0:
10478 0000F78C 66AD                <1>       lodsw
10479 0000F78E C1E010              <1>       shl    eax, 16
10480 0000F791 AC                  <1>       lodsb
10481 0000F792 C1C010              <1>       rol    eax, 16
10482 0000F795 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
10483 0000F79B 7413                <1>       je     short m_pix_op_new_24_1 ; exclude
```

```
10484 0000F79D 668917              <1>       mov    [edi], dx
10485 0000F7A0 C1CA10              <1>       ror    edx, 16
10486 0000F7A3 885702              <1>       mov    [edi+2], dl
10487 0000F7A6 C1C210              <1>       rol    edx, 16
10488 0000F7A9 8305[64030300]03    <1>       add    dword [u.r0], 3
10489                              <1> m_pix_op_new_24_1:
10490 0000F7B0 83C703              <1>       add    edi, 3
10491 0000F7B3 E2D7                <1>       loop   m_pix_op_new_24_0
10492 0000F7B5 C3                  <1>       retn
10493                              <1>       ; 65536 colors (16bpp)
10494                              <1> m_pix_op_new_2:
10495                              <1>       ;jmp   short m_pix_op_new_16
10496                              <1> m_pix_op_new_16:
10497                              <1>       ; 16 bit colors (65536 colors)
10498 0000F7B6 89C2                <1>       mov    edx, eax ; new color
10499                              <1> m_pix_op_new_16_0:
10500 0000F7B8 66AD                <1>       lodsw
10501 0000F7BA 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
10502 0000F7C1 740A                <1>       je     short m_pix_op_new_16_1 ; exclude
10503 0000F7C3 668917              <1>       mov    [edi], dx
10504 0000F7C6 8305[64030300]02    <1>       add    dword [u.r0], 2
10505                              <1> m_pix_op_new_16_1:
10506 0000F7CD 83C702              <1>       add    edi, 2
10507 0000F7D0 E2E6                <1>       loop   m_pix_op_new_16_0
10508 0000F7D2 C3                  <1>       retn
10509                              <1> m_pix_op_new_3:
10510                              <1>       ; 32 bit true colors
10511                              <1>       ;jmp   short m_pix_op_new_32
10512                              <1> m_pix_op_new_32:
10513                              <1>       ; 32 bit true colors
10514 0000F7D3 89C2                <1>       mov    edx, eax ; new color
10515                              <1> m_pix_op_new_32_0:
10516 0000F7D5 AD                  <1>       lodsd
10517 0000F7D6 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
10518 0000F7DC 7409                <1>       je     short m_pix_op_new_32_1 ; exclude
10519 0000F7DE 8917                <1>       mov    [edi], edx
10520 0000F7E0 8305[64030300]04    <1>       add    dword [u.r0], 4
10521                              <1> m_pix_op_new_32_1:
10522 0000F7E7 83C704              <1>       add    edi, 4
10523 0000F7EA E2E9                <1>       loop   m_pix_op_new_32_0
10524 0000F7EC C3                  <1>       retn
10525                              <1>
10526                              <1> m_pix_op_new_w:
10527                              <1>       ; 06/02/2021
10528                              <1>       ; CHANGE COLOR (MASKED, window)
10529                              <1>       ;
10530                              <1>       ; jump from pix_op_new_w
10531                              <1>       ;
10532                              <1>       ; INPUT:
10533                              <1>       ;   ecx = bytes per row (to be applied)
10534                              <1>       ;   edx = screen width in bytes
10535                              <1>       ;   ebx = row count
10536                              <1>       ;   eax = color
10537                              <1>       ;
10538                              <1>       ;   [maskcolor] = mask color (to be excluded)
10539                              <1>       ;
10540                              <1>       ; OUTPUT:
10541                              <1>       ;      [u.r0] will be > 0 if succesful
10542                              <1>
10543                              <1>       ; Window
10544                              <1>       ;mov   edi, [v_str] ; LFB start address
10545                              <1>       ;mov   esi, edi
10546                              <1>
10547 0000F7ED 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
10548 0000F7F4 7707                <1>       ja     short m_pix_op_new_w_1
10549                              <1>
10550                              <1>       ; 256 colors (8bpp)
10551 0000F7F6 BD[68F70000]        <1>       mov    ebp, m_pix_op_new_8
10552 0000F7FB EB1E                <1>       jmp    short m_pix_op_new_w_x
10553                              <1>
10554                              <1> m_pix_op_new_w_1:
10555 0000F7FD 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
10556 0000F804 7710                <1>       ja     short m_pix_op_new_w_3 ; 32bpp
10557 0000F806 7207                <1>       jb     short m_pix_op_new_w_2 ; 16bpp
10558                              <1>
10559                              <1>       ; 24 bit true colors
10560 0000F808 BD[8AF70000]        <1>       mov    ebp, m_pix_op_new_24
10561 0000F80D EB0C                <1>       jmp    short m_pix_op_new_w_x
10562                              <1>
10563                              <1>       ; 65536 colors (16bpp)
10564                              <1> m_pix_op_new_w_2:
10565 0000F80F BD[B6F70000]        <1>       mov    ebp, m_pix_op_new_16
10566 0000F814 EB05                <1>       jmp    short m_pix_op_new_w_x
10567                              <1>
10568                              <1>       ; 32 bit true colors
10569                              <1> m_pix_op_new_w_3:
10570 0000F816 BD[D3F70000]        <1>       mov    ebp, m_pix_op_new_32
10571                              <1>       ;jmp   short m_pix_op_new_w_x
10572                              <1>
10573                              <1> m_pix_op_new_w_x:
10574                              <1> m_pix_op_add_w_x:
10575                              <1> m_pix_op_sub_w_x:
10576                              <1> m_pix_op_mix_w_x:
10577                              <1> m_pix_op_and_w_x:
10578                              <1> m_pix_op_orc_w_x:
10579                              <1> m_pix_op_xor_w_x:
10580                              <1> m_pix_op_not_w_x:
10581                              <1> m_pix_op_neg_w_x:
10582                              <1> m_pix_op_inc_w_x:
10583                              <1> m_pix_op_dec_w_x:
10584                              <1>       ; 27/02/2021
10585                              <1>       ; 26/02/2021
10586                              <1>       ; 06/02/2021
10587                              <1>       ; ecx = bytes per row (to be applied)
10588                              <1>       ; edx = windows (screen) width in bytes
```

```
10589                                  <1>       ; ebx = row count
10590                                  <1>       ; eax = color
10591                                  <1>       ; ebp = pixel operation subroutine address
10592                                  <1>       ; edi = esi = window start address
10593                                  <1>
10594 0000F81B 8B3D[92120300]          <1>       mov    edi, [v_str] ; LFB start address
10595 0000F821 89FE                    <1>       mov    esi, edi
10596                                  <1> m_pix_op_w_x_next:
10597 0000F823 52                      <1>       push   edx
10598 0000F824 51                      <1>       push   ecx
10599 0000F825 56                      <1>       push   esi
10600 0000F826 57                      <1>       push   edi
10601 0000F827 50                      <1>       push   eax ; 26/02/2021
10602 0000F828 8B0D[9E120300]          <1>       mov    ecx, [pixcount] ; 27/02/2021
10603 0000F82E FFD5                    <1>       call   ebp ; call masked pixel-row operation
10604 0000F830 58                      <1>       pop    eax ; 26/02/2021
10605 0000F831 5F                      <1>       pop    edi
10606 0000F832 5E                      <1>       pop    esi
10607 0000F833 59                      <1>       pop    ecx
10608 0000F834 5A                      <1>       pop    edx
10609 0000F835 01D6                    <1>       add    esi, edx ; next row
10610 0000F837 01D7                    <1>       add    edi, edx ; next row
10611 0000F839 4B                      <1>       dec    ebx
10612 0000F83A 75E7                    <1>       jnz    short m_pix_op_w_x_next
10613 0000F83C C3                      <1>       retn
10614                                  <1>
10615                                  <1> m_pix_op_add:
10616                                  <1>       ; 06/02/2021
10617                                  <1>       ; ADD COLOR (MASKED, full screen)
10618                                  <1>       ;
10619                                  <1>       ; jump from pix_op_add
10620                                  <1>       ;
10621                                  <1>       ; INPUT:
10622                                  <1>       ;   eax = color (AL, AX, EAX)
10623                                  <1>       ;   ecx = [v_siz] ; display page pixel count
10624                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
10625                                  <1>       ;
10626                                  <1>       ;   [maskcolor] = mask color (to be excluded)
10627                                  <1>       ;
10628                                  <1>       ; OUTPUT:
10629                                  <1>       ;      [u.r0] will be > 0 if succesful
10630                                  <1>
10631                                  <1>       ; Full screen
10632                                  <1> m_pix_op_add_0:
10633 0000F83D 803D[89120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
10634 0000F844 771C                    <1>       ja     short m_pix_op_add_1
10635                                  <1>       ; 256 colors (8bpp)
10636                                  <1>       ;jmp   short m_pix_op_add_8
10637                                  <1> m_pix_op_add_8:
10638                                  <1>       ; 8 bit colors (256 colors)
10639 0000F846 88C2                    <1>       mov    dl, al ; new color
10640                                  <1> m_pix_op_add_8_0:
10641 0000F848 AC                      <1>       lodsb
10642 0000F849 3A05[9A120300]          <1>       cmp    al, [maskcolor]
10643 0000F84F 740D                    <1>       je     short m_pix_op_add_8_1 ; exclude
10644 0000F851 FF05[64030300]          <1>       inc    dword [u.r0] ; +1
10645 0000F857 0017                    <1>       add    [edi], dl
10646 0000F859 7303                    <1>       jnc    short m_pix_op_add_8_1
10647 0000F85B C607FF                  <1>       mov    byte [edi], 0FFh
10648                                  <1> m_pix_op_add_8_1:
10649 0000F85E 47                      <1>       inc    edi
10650 0000F85F E2E7                    <1>       loop   m_pix_op_add_8_0
10651 0000F861 C3                      <1>       retn
10652                                  <1> m_pix_op_add_1:
10653 0000F862 803D[89120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
10654 0000F869 775E                    <1>       ja     short m_pix_op_add_3 ; 32bpp
10655 0000F86B 7238                    <1>       jb     short m_pix_op_add_2 ; 16bpp
10656                                  <1>       ; 24 bit true colors
10657                                  <1>       ;jmp   short m_pix_op_add_24
10658                                  <1> m_pix_op_add_24:
10659                                  <1>       ; 24 bit true colors
10660 0000F86D 89C2                    <1>       mov    edx, eax ; new color
10661 0000F86F 81CA000000FF            <1>       or     edx, 0FF000000h
10662                                  <1> m_pix_op_add_24_0:
10663 0000F875 66AD                    <1>       lodsw
10664 0000F877 C1E010                  <1>       shl    eax, 16
10665 0000F87A AC                      <1>       lodsb
10666 0000F87B C1C010                  <1>       rol    eax, 16
10667 0000F87E 3B05[9A120300]          <1>       cmp    eax, [maskcolor]
10668 0000F884 7419                    <1>       je     short m_pix_op_add_24_2 ; exclude
10669 0000F886 8305[64030300]03        <1>       add    dword [u.r0], 3 ; +3
10670 0000F88D 01D0                    <1>       add    eax, edx
10671 0000F88F 7305                    <1>       jnc    short m_pix_op_add_24_1
10672 0000F891 B8FFFFFF00              <1>       mov    eax, 0FFFFFFh
10673                                  <1> m_pix_op_add_24_1:
10674 0000F896 668907                  <1>       mov    [edi], ax
10675 0000F899 C1E810                  <1>       shr    eax, 16
10676 0000F89C 884702                  <1>       mov    [edi+2], al
10677                                  <1> m_pix_op_add_24_2:
10678 0000F89F 83C703                  <1>       add    edi, 3 ; +3
10679 0000F8A2 E2D1                    <1>       loop   m_pix_op_add_24_0
10680 0000F8A4 C3                      <1>       retn
10681                                  <1>       ; 65536 colors (16bpp)
10682                                  <1> m_pix_op_add_2:
10683                                  <1>       ;jmp   short m_pix_op_add_16
10684                                  <1> m_pix_op_add_16:
10685                                  <1>       ; 16 bit colors (65536 colors)
10686 0000F8A5 89C2                    <1>       mov    edx, eax ; new color
10687                                  <1> m_pix_op_add_16_0:
10688 0000F8A7 66AD                    <1>       lodsw
10689 0000F8A9 663B05[9A120300]        <1>       cmp    ax, [maskcolor]
10690 0000F8B0 7411                    <1>       je     short m_pix_op_add_16_1 ; exclude
10691 0000F8B2 8305[64030300]02        <1>       add    dword [u.r0], 2 ; +2
10692 0000F8B9 660117                  <1>       add    [edi], dx
10693 0000F8BC 7305                    <1>       jnc    short m_pix_op_add_16_1
```

```
10694 0000F8BE 66C707FFFF          <1>       mov    word [edi], 0FFFFh
10695                              <1> m_pix_op_add_16_1:
10696 0000F8C3 83C702              <1>       add    edi, 2 ; +2
10697 0000F8C6 E2DF                <1>       loop   m_pix_op_add_16_0
10698 0000F8C8 C3                  <1>       retn
10699                              <1> m_pix_op_add_3:
10700                              <1>       ; 32 bit true colors
10701                              <1>       ;jmp   short m_pix_op_add_32
10702                              <1> m_pix_op_add_32:
10703                              <1>       ; 32 bit true colors
10704 0000F8C9 89C2                <1>       mov    edx, eax ; new color
10705                              <1> m_pix_op_add_32_0:
10706 0000F8CB AD                  <1>       lodsd
10707 0000F8CC 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
10708 0000F8D2 7411                <1>       je     short m_pix_op_add_32_1 ; exclude
10709 0000F8D4 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
10710 0000F8DB 0117                <1>       add    [edi], edx
10711 0000F8DD 7306                <1>       jnc    short m_pix_op_add_32_1
10712 0000F8DF C707FFFFFFFF        <1>       mov    dword [edi], 0FFFFFFFFh
10713                              <1> m_pix_op_add_32_1:
10714 0000F8E5 83C704              <1>       add    edi, 4 ; +4
10715 0000F8E8 E2E1                <1>       loop   m_pix_op_add_32_0
10716 0000F8EA C3                  <1>       retn
10717                              <1>
10718                              <1> m_pix_op_add_w:
10719                              <1>       ; 06/02/2021
10720                              <1>       ; ADD COLOR (MASKED, window)
10721                              <1>       ;
10722                              <1>       ; jump from pix_op_add_w
10723                              <1>       ;
10724                              <1>       ; INPUT:
10725                              <1>       ;   ecx = bytes per row (to be applied)
10726                              <1>       ;   edx = screen width in bytes
10727                              <1>       ;   ebx = row count
10728                              <1>       ;   eax = color
10729                              <1>       ;
10730                              <1>       ;   [maskcolor] = mask color (to be excluded)
10731                              <1>       ;
10732                              <1>       ; OUTPUT:
10733                              <1>       ;      [u.r0] will be > 0 if succesful
10734                              <1>
10735                              <1>       ; window
10736                              <1>       ;mov   edi, [v_str] ; LFB start address
10737                              <1>       ;mov   esi, edi
10738                              <1>
10739 0000F8EB 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
10740 0000F8F2 7707                <1>       ja     short m_pix_op_add_w_1
10741                              <1>
10742                              <1>       ; 256 colors (8bpp)
10743 0000F8F4 BD[46F80000]        <1>       mov    ebp, m_pix_op_add_8
10744 0000F8F9 EB1E                <1>       jmp    short m_pix_op_add_w_4
10745                              <1>
10746                              <1> m_pix_op_add_w_1:
10747 0000F8FB 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
10748 0000F902 7710                <1>       ja     short m_pix_op_add_w_3 ; 32bpp
10749 0000F904 7207                <1>       jb     short m_pix_op_add_w_2 ; 16bpp
10750                              <1>
10751                              <1>       ; 24 bit true colors
10752 0000F906 BD[6DF80000]        <1>       mov    ebp, m_pix_op_add_24
10753 0000F90B EB0C                <1>       jmp    short m_pix_op_add_w_4
10754                              <1>
10755                              <1>       ; 65536 colors (16bpp)
10756                              <1> m_pix_op_add_w_2:
10757 0000F90D BD[A5F80000]        <1>       mov    ebp, m_pix_op_add_16
10758 0000F912 EB05                <1>       jmp    short m_pix_op_add_w_4
10759                              <1>
10760                              <1>       ; 32 bit true colors
10761                              <1> m_pix_op_add_w_3:
10762 0000F914 BD[C9F80000]        <1>       mov    ebp, m_pix_op_add_32
10763                              <1> m_pix_op_add_w_4:
10764 0000F919 E9FDFEFFFF          <1>       jmp    m_pix_op_add_w_x
10765                              <1>
10766                              <1> m_pix_op_sub:
10767                              <1>       ; 02/03/2021
10768                              <1>       ; 06/02/2021
10769                              <1>       ; SUBTRACT COLOR (MASKED, full screen)
10770                              <1>       ;
10771                              <1>       ; jump from pix_op_sub
10772                              <1>       ;
10773                              <1>       ; INPUT:
10774                              <1>       ;   eax = color (AL, AX, EAX)
10775                              <1>       ;   ecx = [v_siz] ; display page pixel count
10776                              <1>       ;   esi = edi = [v_mem] ; LFB start address
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>       ; Full screen
10784                              <1> m_pix_op_sub_0:
10785 0000F91E 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
10786 0000F925 771C                <1>       ja     short m_pix_op_sub_1
10787                              <1>       ; 256 colors (8bpp)
10788                              <1>       ;jmp   short m_pix_op_sub_8
10789                              <1> m_pix_op_sub_8:
10790                              <1>       ; 8 bit colors (256 colors)
10791 0000F927 88C2                <1>       mov    dl, al ; new color
10792                              <1> m_pix_op_sub_8_0:
10793 0000F929 AC                  <1>       lodsb
10794 0000F92A 3A05[9A120300]      <1>       cmp    al, [maskcolor]
10795 0000F930 740D                <1>       je     short m_pix_op_sub_8_1 ; exclude
10796 0000F932 FF05[64030300]      <1>       inc    dword [u.r0] ; +1
10797 0000F938 2817                <1>       sub    [edi], dl
10798 0000F93A 7303                <1>       jnb    short m_pix_op_sub_8_1
```

```
10799 0000F93C C60700              <1>       mov   byte [edi], 0
10800                              <1> m_pix_op_sub_8_1:
10801 0000F93F 47                  <1>       inc   edi
10802 0000F940 E2E7                <1>       loop  m_pix_op_sub_8_0
10803 0000F942 C3                  <1>       retn
10804                              <1> m_pix_op_sub_1:
10805 0000F943 803D[89120300]18    <1>       cmp   byte [v_bpp], 24 ; 24bpp
10806 0000F94A 7755                <1>       ja    short m_pix_op_sub_3 ; 32bpp
10807 0000F94C 722F                <1>       jb    short m_pix_op_sub_2 ; 16bpp
10808                              <1>       ; 24 bit true colors
10809                              <1>       ;jmp  short m_pix_op_sub_24
10810                              <1> m_pix_op_sub_24:
10811                              <1>       ; 24 bit true colors
10812 0000F94E 89C2                <1>       mov   edx, eax ; new color
10813                              <1>       ; 02/03/2021
10814                              <1> m_pix_op_sub_24_0:
10815 0000F950 66AD                <1>       lodsw
10816 0000F952 C1E010              <1>       shl   eax, 16
10817 0000F955 AC                  <1>       lodsb
10818 0000F956 C1C010              <1>       rol   eax, 16
10819 0000F959 3B05[9A120300]      <1>       cmp   eax, [maskcolor]
10820 0000F95F 7416                <1>       je    short m_pix_op_sub_24_2 ; exclude
10821 0000F961 8305[64030300]03    <1>       add   dword [u.r0], 3 ; +3
10822 0000F968 29D0                <1>       sub   eax, edx
10823 0000F96A 7302                <1>       jnb   short m_pix_op_sub_24_1
10824 0000F96C 31C0                <1>       xor   eax, eax ; 0
10825                              <1> m_pix_op_sub_24_1:
10826 0000F96E 668907              <1>       mov   [edi], ax
10827 0000F971 C1E810              <1>       shr   eax, 16
10828 0000F974 884702              <1>       mov   [edi+2], al
10829                              <1> m_pix_op_sub_24_2:
10830 0000F977 83C703              <1>       add   edi, 3 ; +3
10831 0000F97A E2D4                <1>       loop  m_pix_op_sub_24_0
10832 0000F97C C3                  <1>       retn
10833                              <1>       ; 65536 colors (16bpp)
10834                              <1> m_pix_op_sub_2:
10835                              <1>       ;jmp  short m_pix_op_sub_16
10836                              <1> m_pix_op_sub_16:
10837                              <1>       ; 16 bit colors (65536 colors)
10838 0000F97D 89C2                <1>       mov   edx, eax ; new color
10839                              <1> m_pix_op_sub_16_0:
10840 0000F97F 66AD                <1>       lodsw
10841 0000F981 663B05[9A120300]    <1>       cmp   ax, [maskcolor]
10842 0000F988 7411                <1>       je    short m_pix_op_sub_16_1 ; exclude
10843 0000F98A 8305[64030300]02    <1>       add   dword [u.r0], 2 ; +2
10844 0000F991 662917              <1>       sub   [edi], dx
10845 0000F994 7305                <1>       jnb   short m_pix_op_sub_16_1
10846 0000F996 31C0                <1>       xor   eax, eax
10847 0000F998 668907              <1>       mov   [edi], ax ; 0
10848                              <1> m_pix_op_sub_16_1:
10849 0000F99B 83C702              <1>       add   edi, 2 ; +2
10850 0000F99E E2DF                <1>       loop  m_pix_op_sub_16_0
10851 0000F9A0 C3                  <1>       retn
10852                              <1> m_pix_op_sub_3:
10853                              <1>       ; 32 bit true colors
10854                              <1>       ;jmp  short m_pix_op_sub_32
10855                              <1> m_pix_op_sub_32:
10856                              <1>       ; 32 bit true colors
10857 0000F9A1 89C2                <1>       mov   edx, eax ; new color
10858                              <1> m_pix_op_sub_32_0:
10859 0000F9A3 AD                  <1>       lodsd
10860 0000F9A4 3B05[9A120300]      <1>       cmp   eax, [maskcolor]
10861 0000F9AA 740F                <1>       je    short m_pix_op_sub_32_1 ; exclude
10862 0000F9AC 8305[64030300]04    <1>       add   dword [u.r0], 4 ; +4
10863 0000F9B3 2917                <1>       sub   [edi], edx
10864 0000F9B5 7304                <1>       jnb   short m_pix_op_sub_32_1
10865 0000F9B7 31C0                <1>       xor   eax, eax
10866 0000F9B9 8907                <1>       mov   [edi], eax ; 0
10867                              <1> m_pix_op_sub_32_1:
10868 0000F9BB 83C704              <1>       add   edi, 4 ; +4
10869 0000F9BE E2E3                <1>       loop  m_pix_op_sub_32_0
10870 0000F9C0 C3                  <1>       retn
10871                              <1>
10872                              <1> m_pix_op_sub_w:
10873                              <1>       ; 06/02/2021
10874                              <1>       ; SUBTRACT COLOR (MASKED, window)
10875                              <1>       ;
10876                              <1>       ; jump from pix_op_sub_w
10877                              <1>       ;
10878                              <1>       ; INPUT:
10879                              <1>       ;   ecx = bytes per row (to be applied)
10880                              <1>       ;   edx = screen width in bytes
10881                              <1>       ;   ebx = row count
10882                              <1>       ;   eax = color
10883                              <1>       ;
10884                              <1>       ;   [maskcolor] = mask color (to be excluded)
10885                              <1>       ;
10886                              <1>       ; OUTPUT:
10887                              <1>       ;     [u.r0] will be > 0 if succesful
10888                              <1>
10889                              <1>       ; window
10890                              <1>       ;mov  edi, [v_str] ; LFB start address
10891                              <1>       ;mov  esi, edi
10892                              <1>
10893 0000F9C1 803D[89120300]08    <1>       cmp   byte [v_bpp], 8 ; 8bpp
10894 0000F9C8 7707                <1>       ja    short m_pix_op_sub_w_1
10895                              <1>
10896                              <1>       ; 256 colors (8bpp)
10897 0000F9CA BD[27F90000]        <1>       mov   ebp, m_pix_op_sub_8
10898 0000F9CF EB1E                <1>       jmp   short m_pix_op_sub_w_4
10899                              <1>
10900                              <1> m_pix_op_sub_w_1:
10901 0000F9D1 803D[89120300]18    <1>       cmp   byte [v_bpp], 24 ; 24bpp
10902 0000F9D8 7710                <1>       ja    short m_pix_op_sub_w_3 ; 32bpp
10903 0000F9DA 7207                <1>       jb    short m_pix_op_sub_w_2 ; 16bpp
```

```
                                <1>
                                <1>       ; 24 bit true colors
0000F9DC BD[4EF90000]           <1>       mov    ebp, m_pix_op_sub_24
0000F9E1 EB0C                   <1>       jmp    short m_pix_op_sub_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_sub_w_2:
0000F9E3 BD[7DF90000]           <1>       mov    ebp, m_pix_op_sub_16
0000F9E8 EB05                   <1>       jmp    short m_pix_op_sub_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> m_pix_op_sub_w_3:
0000F9EA BD[A1F90000]           <1>       mov    ebp, m_pix_op_sub_32
                                <1> m_pix_op_sub_w_4:
0000F9EF E927FEFFFF             <1>       jmp    m_pix_op_sub_w_x
                                <1>
                                <1> m_pix_op_mix:
                                <1>       ; 25/02/2021
                                <1>       ; 06/02/2021
                                <1>       ; MIX COLOR (MASKED, full screen)
                                <1>       ;
                                <1>       ; jump from pix_op_mix
                                <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_mix_0:
0000F9F4 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000F9FB 771B                   <1>       ja     short m_pix_op_mix_1
                                <1>       ; 256 colors (8bpp)
                                <1>       ;jmp   short m_pix_op_mix_8
                                <1> m_pix_op_mix_8:
                                <1>       ; 8 bit colors (256 colors)
0000F9FD 88C2                   <1>       mov    dl, al ; new (mixing) color
                                <1> m_pix_op_mix_8_0:
0000F9FF AC                     <1>       lodsb
0000FA00 3A05[9A120300]         <1>       cmp    al, [maskcolor]
0000FA06 740C                   <1>       je     short m_pix_op_mix_8_1 ; exclude
0000FA08 00D0                   <1>       add    al, dl ; 25/02/2021
0000FA0A D0D8                   <1>       rcr    al, 1
0000FA0C 8807                   <1>       mov    [edi], al
0000FA0E FF05[64030300]         <1>       inc    dword [u.r0] ; +1
                                <1> m_pix_op_mix_8_1:
0000FA14 47                     <1>       inc    edi
0000FA15 E2E8                   <1>       loop   m_pix_op_mix_8_0
0000FA17 C3                     <1>       retn
                                <1> m_pix_op_mix_1:
0000FA18 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FA1F 7752                   <1>       ja     short m_pix_op_mix_3 ; 32bpp
0000FA21 722D                   <1>       jb     short m_pix_op_mix_2 ; 16bpp
                                <1>       ; 24 bit true colors
                                <1>       ;jmp   short m_pix_op_mix_24
                                <1> m_pix_op_mix_24:
                                <1>       ; 24 bit true colors
0000FA23 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFFFh
                                <1> m_pix_op_mix_24_0:
0000FA25 66AD                   <1>       lodsw
0000FA27 C1E010                 <1>       shl    eax, 16
0000FA2A AC                     <1>       lodsb
0000FA2B C1C010                 <1>       rol    eax, 16
0000FA2E 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FA34 7414                   <1>       je     short m_pix_op_mix_24_1 ; exclude
0000FA36 01D0                   <1>       add    eax, edx
0000FA38 D1E8                   <1>       shr    eax, 1
0000FA3A 668907                 <1>       mov    [edi], ax
0000FA3D C1E810                 <1>       shr    eax, 16
0000FA40 884702                 <1>       mov    [edi+2], al
0000FA43 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_mix_24_1:
0000FA4A 83C703                 <1>       add    edi, 3 ; +3
0000FA4D E2D6                   <1>       loop   m_pix_op_mix_24_0
0000FA4F C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_mix_2:
                                <1>       ;jmp   short m_pix_op_mix_16
                                <1> m_pix_op_mix_16:
                                <1>       ; 16 bit colors (65536 colors)
0000FA50 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFh
                                <1> m_pix_op_mix_16_0:
0000FA52 66AD                   <1>       lodsw
0000FA54 663B05[9A120300]       <1>       cmp    ax, [maskcolor]
0000FA5B 7410                   <1>       je     short m_pix_op_mix_16_1 ; exclude
0000FA5D 6601D0                 <1>       add    ax, dx
0000FA60 66D1D8                 <1>       rcr    ax, 1
0000FA63 668907                 <1>       mov    [edi], ax
0000FA66 8305[64030300]02       <1>       add    dword [u.r0], 2 ; +2
                                <1> m_pix_op_mix_16_1:
0000FA6D 83C702                 <1>       add    edi, 2 ; +2
0000FA70 E2E0                   <1>       loop   m_pix_op_mix_16_0
0000FA72 C3                     <1>       retn
                                <1> m_pix_op_mix_3:
                                <1>       ; 32 bit true colors
                                <1>       ;jmp   short m_pix_op_mix_32
                                <1> m_pix_op_mix_32:
                                <1>       ; 32 bit true colors
```

```
11009 0000FA73 89C2                <1>       mov    edx, eax ; new color
11010                              <1> m_pix_op_mix_32_0:
11011 0000FA75 AD                  <1>       lodsd
11012 0000FA76 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
11013 0000FA7C 740D                <1>       je     short m_pix_op_mix_32_1 ; exclude
11014 0000FA7E 01D0                <1>       add    eax, edx
11015                              <1>       ; 02/03/2021
11016 0000FA80 D1D8                <1>       rcr    eax, 1
11017 0000FA82 8907                <1>       mov    [edi], eax
11018 0000FA84 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
11019                              <1> m_pix_op_mix_32_1:
11020 0000FA8B 83C704              <1>       add    edi, 4 ; +4
11021 0000FA8E E2E5                <1>       loop   m_pix_op_mix_32_0
11022 0000FA90 C3                  <1>       retn
11023                              <1>
11024                              <1> m_pix_op_mix_w:
11025                              <1>       ; 06/02/2021
11026                              <1>       ; MIX COLOR (MASKED, window)
11027                              <1>       ;
11028                              <1>       ; jump from pix_op_mix_w
11029                              <1>       ;
11030                              <1>       ; INPUT:
11031                              <1>       ;   ecx = bytes per row (to be applied)
11032                              <1>       ;   edx = screen width in bytes
11033                              <1>       ;   ebx = row count
11034                              <1>       ;   eax = color
11035                              <1>       ;
11036                              <1>       ;   [maskcolor] = mask color (to be excluded)
11037                              <1>       ;
11038                              <1>       ; OUTPUT:
11039                              <1>       ;      [u.r0] will be > 0 if succesful
11040                              <1>
11041                              <1>       ; window
11042                              <1>       ;mov   edi, [v_str] ; LFB start address
11043                              <1>       ;mov   esi, edi
11044                              <1>
11045 0000FA91 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11046 0000FA98 7707                <1>       ja     short m_pix_op_mix_w_1
11047                              <1>
11048                              <1>       ; 256 colors (8bpp)
11049 0000FA9A BD[FDF90000]        <1>       mov    ebp, m_pix_op_mix_8
11050 0000FA9F EB1E                <1>       jmp    short m_pix_op_mix_w_4
11051                              <1>
11052                              <1> m_pix_op_mix_w_1:
11053 0000FAA1 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11054 0000FAA8 7710                <1>       ja     short m_pix_op_mix_w_3 ; 32bpp
11055 0000FAAA 7207                <1>       jb     short m_pix_op_mix_w_2 ; 16bpp
11056                              <1>
11057                              <1>       ; 24 bit true colors
11058 0000FAAC BD[23FA0000]        <1>       mov    ebp, m_pix_op_mix_24
11059 0000FAB1 EB0C                <1>       jmp    short m_pix_op_mix_w_4
11060                              <1>
11061                              <1>       ; 65536 colors (16bpp)
11062                              <1> m_pix_op_mix_w_2:
11063 0000FAB3 BD[50FA0000]        <1>       mov    ebp, m_pix_op_mix_16
11064 0000FAB8 EB05                <1>       jmp    short m_pix_op_mix_w_4
11065                              <1>
11066                              <1>       ; 32 bit true colors
11067                              <1> m_pix_op_mix_w_3:
11068 0000FABA BD[73FA0000]        <1>       mov    ebp, m_pix_op_mix_32
11069                              <1> m_pix_op_mix_w_4:
11070 0000FABF E957FDFFFF          <1>       jmp    m_pix_op_mix_w_x
11071                              <1>
11072                              <1> m_pix_op_and:
11073                              <1>       ; 06/02/2021
11074                              <1>       ; AND COLOR (MASKED, full screen)
11075                              <1>       ;
11076                              <1>       ; jump from pix_op_and
11077                              <1>       ;
11078                              <1>       ; INPUT:
11079                              <1>       ;   eax = color (AL, AX, EAX)
11080                              <1>       ;   ecx = [v_siz] ; display page pixel count
11081                              <1>       ;   esi = edi = [v_mem] ; LFB start address
11082                              <1>       ;
11083                              <1>       ;   [maskcolor] = mask color (to be excluded)
11084                              <1>       ;
11085                              <1>       ; OUTPUT:
11086                              <1>       ;      [u.r0] will be > 0 if succesful
11087                              <1>
11088                              <1>       ; Full screen
11089                              <1> m_pix_op_and_0:
11090 0000FAC4 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11091 0000FACB 7717                <1>       ja     short m_pix_op_and_1
11092                              <1>       ; 256 colors (8bpp)
11093                              <1>       ;jmp   short m_pix_op_and_8
11094                              <1> m_pix_op_and_8:
11095                              <1>       ; 8 bit colors (256 colors)
11096 0000FACD 88C2                <1>       mov    dl, al ; new color
11097                              <1> m_pix_op_and_8_0:
11098 0000FACF AC                  <1>       lodsb
11099 0000FAD0 3A05[9A120300]      <1>       cmp    al, [maskcolor]
11100 0000FAD6 7408                <1>       je     short m_pix_op_and_8_1 ; exclude
11101 0000FAD8 2017                <1>       and    [edi], dl
11102 0000FADA FF05[64030300]      <1>       inc    dword [u.r0] ; +1
11103                              <1> m_pix_op_and_8_1:
11104 0000FAE0 47                  <1>       inc    edi
11105 0000FAE1 E2EC                <1>       loop   m_pix_op_and_8_0
11106 0000FAE3 C3                  <1>       retn
11107                              <1> m_pix_op_and_1:
11108 0000FAE4 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11109 0000FAEB 774A                <1>       ja     short m_pix_op_and_3 ; 32bpp
11110 0000FAED 722B                <1>       jb     short m_pix_op_and_2 ; 16bpp
11111                              <1>       ; 24 bit true colors
11112                              <1>       ;jmp   short m_pix_op_and_24
11113                              <1> m_pix_op_and_24:
```

```
                                <1>       ; 24 bit true colors
0000FAEF 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFFFh
                                <1> m_pix_op_and_24_0:
0000FAF1 66AD                   <1>       lodsw
0000FAF3 C1E010                 <1>       shl    eax, 16
0000FAF6 AC                     <1>       lodsb
0000FAF7 C1C010                 <1>       rol    eax, 16
0000FAFA 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FB00 7412                   <1>       je     short m_pix_op_and_24_1 ; exclude
0000FB02 21D0                   <1>       and    eax, edx
0000FB04 668907                 <1>       mov    [edi], ax
0000FB07 C1E810                 <1>       shr    eax, 16
0000FB0A 884702                 <1>       mov    [edi+2], al
0000FB0D 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_and_24_1:
0000FB14 83C703                 <1>       add    edi, 3 ; +3
0000FB17 E2D8                   <1>       loop   m_pix_op_and_24_0
0000FB19 C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_and_2:
                                <1>       ;jmp   short m_pix_op_and_16
                                <1> m_pix_op_and_16:
                                <1>       ; 16 bit colors (65536 colors)
0000FB1A 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFh
                                <1> m_pix_op_and_16_0:
0000FB1C 66AD                   <1>       lodsw
0000FB1E 663B05[9A120300]       <1>       cmp    ax, [maskcolor]
0000FB25 740A                   <1>       je     short m_pix_op_and_16_1 ; exclude
0000FB27 662117                 <1>       and    [edi], dx
0000FB2A 8305[64030300]02       <1>       add    dword [u.r0], 2 ; +2
                                <1> m_pix_op_and_16_1:
0000FB31 83C702                 <1>       add    edi, 2 ; +2
0000FB34 E2E6                   <1>       loop   m_pix_op_and_16_0
0000FB36 C3                     <1>       retn
                                <1> m_pix_op_and_3:
                                <1>       ; 32 bit true colors
                                <1>       ;jmp   short m_pix_op_and_32
                                <1> m_pix_op_and_32:
                                <1>       ; 32 bit true colors
0000FB37 89C2                   <1>       mov    edx, eax ; new color
                                <1> m_pix_op_and_32_0:
0000FB39 AD                     <1>       lodsd
0000FB3A 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FB40 7409                   <1>       je     short m_pix_op_and_32_1 ; exclude
0000FB42 2117                   <1>       and    [edi], edx ; 25/02/2021
0000FB44 8305[64030300]04       <1>       add    dword [u.r0], 4 ; +4
                                <1> m_pix_op_and_32_1:
0000FB4B 83C704                 <1>       add    edi, 4 ; +4
0000FB4E E2E9                   <1>       loop   m_pix_op_and_32_0
0000FB50 C3                     <1>       retn
                                <1>
                                <1> m_pix_op_and_w:
                                <1>       ; 06/02/2021
                                <1>       ; AND COLOR (MASKED, window)
                                <1>       ;
                                <1>       ; jump from pix_op_and_w
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;   ecx = bytes per row (to be applied)
                                <1>       ;   edx = screen width in bytes
                                <1>       ;   ebx = row count
                                <1>       ;   eax = color
                                <1>       ;
                                <1>       ;   [maskcolor] = mask color (to be excluded)
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
                                <1>       ; window
                                <1>       ;mov   edi, [v_str] ; LFB start address
                                <1>       ;mov   esi, edi
                                <1>
0000FB51 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000FB58 7707                   <1>       ja     short m_pix_op_and_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000FB5A BD[CDFA0000]           <1>       mov    ebp, m_pix_op_and_8
0000FB5F EB1E                   <1>       jmp    short m_pix_op_and_w_4
                                <1>
                                <1> m_pix_op_and_w_1:
0000FB61 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FB68 7710                   <1>       ja     short m_pix_op_and_w_3 ; 32bpp
0000FB6A 7207                   <1>       jb     short m_pix_op_and_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000FB6C BD[EFFA0000]           <1>       mov    ebp, m_pix_op_and_24
0000FB71 EB0C                   <1>       jmp    short m_pix_op_and_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_and_w_2:
0000FB73 BD[1AFB0000]           <1>       mov    ebp, m_pix_op_and_16
0000FB78 EB05                   <1>       jmp    short m_pix_op_and_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> m_pix_op_and_w_3:
0000FB7A BD[37FB0000]           <1>       mov    ebp, m_pix_op_and_32
                                <1> m_pix_op_and_w_4:
0000FB7F 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>       ;
```

```
11219                                  <1>       ; jump from pix_op_orc
11220                                  <1>       ;
11221                                  <1>       ; INPUT:
11222                                  <1>       ;   eax = color (AL, AX, EAX)
11223                                  <1>       ;   ecx = [v_siz] ; display page pixel count
11224                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
11225                                  <1>       ;
11226                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11227                                  <1>       ;
11228                                  <1>       ; OUTPUT:
11229                                  <1>       ;      [u.r0] will be > 0 if succesful
11230                                  <1>
11231                                  <1>       ; Full screen
11232                                  <1> m_pix_op_or_0:
11233 0000FB84 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
11234 0000FB8B 7717                   <1>       ja     short m_pix_op_or_1
11235                                  <1>       ; 256 colors (8bpp)
11236                                  <1>       ;jmp   short m_pix_op_or_8
11237                                  <1> m_pix_op_or_8:
11238                                  <1>       ; 8 bit colors (256 colors)
11239 0000FB8D 88C2                   <1>       mov    dl, al ; new color
11240                                  <1> m_pix_op_or_8_0:
11241 0000FB8F AC                     <1>       lodsb
11242 0000FB90 3A05[9A120300]         <1>       cmp    al, [maskcolor]
11243 0000FB96 7408                   <1>       je     short m_pix_op_or_8_1 ; exclude
11244 0000FB98 0817                   <1>       or     [edi], dl
11245 0000FB9A FF05[64030300]         <1>       inc    dword [u.r0] ; +1
11246                                  <1> m_pix_op_or_8_1:
11247 0000FBA0 47                     <1>       inc    edi
11248 0000FBA1 E2EC                   <1>       loop   m_pix_op_or_8_0
11249 0000FBA3 C3                     <1>       retn
11250                                  <1> m_pix_op_or_1:
11251 0000FBA4 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
11252 0000FBAB 774A                   <1>       ja     short m_pix_op_or_3 ; 32bpp
11253 0000FBAD 722B                   <1>       jb     short m_pix_op_or_2 ; 16bpp
11254                                  <1>       ; 24 bit true colors
11255                                  <1>       ;jmp   short m_pix_op_or_24
11256                                  <1> m_pix_op_or_24:
11257                                  <1>       ; 24 bit true colors
11258 0000FBAF 89C2                   <1>       mov    edx, eax ; new color
11259                                  <1>       ;and   edx, 0FFFFFFh
11260                                  <1> m_pix_op_or_24_0:
11261 0000FBB1 66AD                   <1>       lodsw
11262 0000FBB3 C1E010                 <1>       shl    eax, 16
11263 0000FBB6 AC                     <1>       lodsb
11264 0000FBB7 C1C010                 <1>       rol    eax, 16
11265 0000FBBA 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
11266 0000FBC0 7412                   <1>       je     short m_pix_op_or_24_1 ; exclude
11267 0000FBC2 09D0                   <1>       or     eax, edx
11268 0000FBC4 668907                 <1>       mov    [edi], ax
11269 0000FBC7 C1E810                 <1>       shr    eax, 16
11270 0000FBCA 884702                 <1>       mov    [edi+2], al
11271 0000FBCD 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
11272                                  <1> m_pix_op_or_24_1:
11273 0000FBD4 83C703                 <1>       add    edi, 3 ; +3
11274 0000FBD7 E2D8                   <1>       loop   m_pix_op_or_24_0
11275 0000FBD9 C3                     <1>       retn
11276                                  <1>       ; 65536 colors (16bpp)
11277                                  <1> m_pix_op_or_2:
11278                                  <1>       ;jmp   short m_pix_op_or_16
11279                                  <1> m_pix_op_or_16:
11280                                  <1>       ; 16 bit colors (65536 colors)
11281 0000FBDA 89C2                   <1>       mov    edx, eax ; new color
11282                                  <1>       ;and   edx, 0FFFFh
11283                                  <1> m_pix_op_or_16_0:
11284 0000FBDC 66AD                   <1>       lodsw
11285 0000FBDE 663B05[9A120300]       <1>       cmp    ax, [maskcolor]
11286 0000FBE5 740A                   <1>       je     short m_pix_op_or_16_1 ; exclude
11287 0000FBE7 660917                 <1>       or     [edi], dx
11288 0000FBEA 8305[64030300]02       <1>       add    dword [u.r0], 2 ; +2
11289                                  <1> m_pix_op_or_16_1:
11290 0000FBF1 83C702                 <1>       add    edi, 2 ; +2
11291 0000FBF4 E2E6                   <1>       loop   m_pix_op_or_16_0
11292 0000FBF6 C3                     <1>       retn
11293                                  <1> m_pix_op_or_3:
11294                                  <1>       ; 32 bit true colors
11295                                  <1>       ;jmp   short m_pix_op_or_32
11296                                  <1> m_pix_op_or_32:
11297                                  <1>       ; 32 bit true colors
11298 0000FBF7 89C2                   <1>       mov    edx, eax ; new color
11299                                  <1> m_pix_op_or_32_0:
11300 0000FBF9 AD                     <1>       lodsd
11301 0000FBFA 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
11302 0000FC00 7409                   <1>       je     short m_pix_op_or_32_1 ; exclude
11303 0000FC02 0917                   <1>       or     [edi], edx ; 25/02/2021
11304 0000FC04 8305[64030300]04       <1>       add    dword [u.r0], 4 ; +4
11305                                  <1> m_pix_op_or_32_1:
11306 0000FC0B 83C704                 <1>       add    edi, 4 ; +4
11307 0000FC0E E2E9                   <1>       loop   m_pix_op_or_32_0
11308 0000FC10 C3                     <1>       retn
11309                                  <1>
11310                                  <1> m_pix_op_or_w:
11311                                  <1>       ; 06/02/2021
11312                                  <1>       ; MIX COLOR (MASKED, window)
11313                                  <1>       ;
11314                                  <1>       ; jump from pix_op_or_w
11315                                  <1>       ;
11316                                  <1>       ; INPUT:
11317                                  <1>       ;   ecx = bytes per row (to be applied)
11318                                  <1>       ;   edx = screen width in bytes
11319                                  <1>       ;   ebx = row count
11320                                  <1>       ;   eax = color
11321                                  <1>       ;
11322                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11323                                  <1>       ;
```

```
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
                                <1>       ; window
                                <1>       ;mov   edi, [v_str] ; LFB start address
                                <1>       ;mov   esi, edi
                                <1>
0000FC11 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000FC18 7707                   <1>       ja     short m_pix_op_or_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000FC1A BD[8DFB0000]           <1>       mov    ebp, m_pix_op_or_8
0000FC1F EB1E                   <1>       jmp    short m_pix_op_or_w_4
                                <1>
                                <1> m_pix_op_or_w_1:
0000FC21 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FC28 7710                   <1>       ja     short m_pix_op_or_w_3 ; 32bpp
0000FC2A 7207                   <1>       jb     short m_pix_op_or_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000FC2C BD[AFFB0000]           <1>       mov    ebp, m_pix_op_or_24
0000FC31 EB0C                   <1>       jmp    short m_pix_op_or_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_or_w_2:
0000FC33 BD[DAFB0000]           <1>       mov    ebp, m_pix_op_or_16
0000FC38 EB05                   <1>       jmp    short m_pix_op_or_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> m_pix_op_or_w_3:
0000FC3A BD[F7FB0000]           <1>       mov    ebp, m_pix_op_or_32
                                <1> m_pix_op_or_w_4:
0000FC3F E9D7FBFFFF             <1>       jmp    m_pix_op_orc_w_x
                                <1>
                                <1> m_pix_op_xor:
                                <1>       ; 06/02/2021
                                <1>       ; XOR COLOR (MASKED, full screen)
                                <1>       ;
                                <1>       ; jump from pix_op_xor
                                <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_xor_0:
0000FC44 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000FC4B 7717                   <1>       ja     short m_pix_op_xor_1
                                <1>       ; 256 colors (8bpp)
                                <1>       ;jmp   short m_pix_op_xor_8
                                <1> m_pix_op_xor_8:
                                <1>       ; 8 bit colors (256 colors)
0000FC4D 88C2                   <1>       mov    dl, al ; new color
                                <1> m_pix_op_xor_8_0:
0000FC4F AC                     <1>       lodsb
0000FC50 3A05[9A120300]         <1>       cmp    al, [maskcolor]
0000FC56 7408                   <1>       je     short m_pix_op_xor_8_1 ; exclude
0000FC58 3017                   <1>       xor    [edi], dl
0000FC5A FF05[64030300]         <1>       inc    dword [u.r0] ; +1
                                <1> m_pix_op_xor_8_1:
0000FC60 47                     <1>       inc    edi
0000FC61 E2EC                   <1>       loop   m_pix_op_xor_8_0
0000FC63 C3                     <1>       retn
                                <1> m_pix_op_xor_1:
0000FC64 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FC6B 774A                   <1>       ja     short m_pix_op_xor_3 ; 32bpp
0000FC6D 722B                   <1>       jb     short m_pix_op_xor_2 ; 16bpp
                                <1>       ; 24 bit true colors
                                <1>       ;jmp   short m_pix_op_xor_24
                                <1> m_pix_op_xor_24:
                                <1>       ; 24 bit true colors
0000FC6F 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFFFh
                                <1> m_pix_op_xor_24_0:
0000FC71 66AD                   <1>       lodsw
0000FC73 C1E010                 <1>       shl    eax, 16
0000FC76 AC                     <1>       lodsb
0000FC77 C1C010                 <1>       rol    eax, 16
0000FC7A 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FC80 7412                   <1>       je     short m_pix_op_xor_24_1 ; exclude
0000FC82 31D0                   <1>       xor    eax, edx
0000FC84 668907                 <1>       mov    [edi], ax
0000FC87 C1E810                 <1>       shr    eax, 16
0000FC8A 884702                 <1>       mov    [edi+2], al
0000FC8D 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_xor_24_1:
0000FC94 83C703                 <1>       add    edi, 3 ; +3
0000FC97 E2D8                   <1>       loop   m_pix_op_xor_24_0
0000FC99 C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_xor_2:
                                <1>       ;jmp   short m_pix_op_xor_16
                                <1> m_pix_op_xor_16:
                                <1>       ; 16 bit colors (65536 colors)
0000FC9A 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFh
                                <1> m_pix_op_xor_16_0:
0000FC9C 66AD                   <1>       lodsw
0000FC9E 663B05[9A120300]       <1>       cmp    ax, [maskcolor]
```

```
11429 0000FCA5 740A                <1>       je     short m_pix_op_xor_16_1 ; exclude
11430 0000FCA7 663117              <1>       xor    [edi], dx
11431 0000FCAA 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
11432                              <1> m_pix_op_xor_16_1:
11433 0000FCB1 83C702              <1>       add    edi, 2 ; +2
11434 0000FCB4 E2E6                <1>       loop   m_pix_op_xor_16_0
11435 0000FCB6 C3                  <1>       retn
11436                              <1> m_pix_op_xor_3:
11437                              <1>       ; 32 bit true colors
11438                              <1>       ;jmp   short m_pix_op_xor_32
11439                              <1> m_pix_op_xor_32:
11440                              <1>       ; 32 bit true colors
11441 0000FCB7 89C2                <1>       mov    edx, eax ; new color
11442                              <1> m_pix_op_xor_32_0:
11443 0000FCB9 AD                  <1>       lodsd
11444 0000FCBA 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
11445 0000FCC0 7409                <1>       je     short m_pix_op_xor_32_1 ; exclude
11446 0000FCC2 3117                <1>       xor    [edi], edx ; 25/02/2021
11447 0000FCC4 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
11448                              <1> m_pix_op_xor_32_1:
11449 0000FCCB 83C704              <1>       add    edi, 4 ; +4
11450 0000FCCE E2E9                <1>       loop   m_pix_op_xor_32_0
11451 0000FCD0 C3                  <1>       retn
11452                              <1>
11453                              <1> m_pix_op_xor_w:
11454                              <1>       ; 06/02/2021
11455                              <1>       ; XOR COLOR (MASKED, window)
11456                              <1>       ;
11457                              <1>       ; jump from pix_op_xor_w
11458                              <1>       ;
11459                              <1>       ; INPUT:
11460                              <1>       ;   ecx = bytes per row (to be applied)
11461                              <1>       ;   edx = screen width in bytes
11462                              <1>       ;   ebx = row count
11463                              <1>       ;   eax = color
11464                              <1>       ;
11465                              <1>       ;   [maskcolor] = mask color (to be excluded)
11466                              <1>       ;
11467                              <1>       ; OUTPUT:
11468                              <1>       ;      [u.r0] will be > 0 if succesful
11469                              <1>
11470                              <1>       ; window
11471                              <1>       ;mov   edi, [v_str] ; LFB start address
11472                              <1>       ;mov   esi, edi
11473                              <1>
11474 0000FCD1 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11475 0000FCD8 7707                <1>       ja     short m_pix_op_xor_w_1
11476                              <1>
11477                              <1>       ; 256 colors (8bpp)
11478 0000FCDA BD[4DFC0000]        <1>       mov    ebp, m_pix_op_xor_8
11479 0000FCDF EB1E                <1>       jmp    short m_pix_op_xor_w_4
11480                              <1>
11481                              <1> m_pix_op_xor_w_1:
11482 0000FCE1 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11483 0000FCE8 7710                <1>       ja     short m_pix_op_xor_w_3 ; 32bpp
11484 0000FCEA 7207                <1>       jb     short m_pix_op_xor_w_2 ; 16bpp
11485                              <1>
11486                              <1>       ; 24 bit true colors
11487 0000FCEC BD[6FFC0000]        <1>       mov    ebp, m_pix_op_xor_24
11488 0000FCF1 EB0C                <1>       jmp    short m_pix_op_xor_w_4
11489                              <1>
11490                              <1>       ; 65536 colors (16bpp)
11491                              <1> m_pix_op_xor_w_2:
11492 0000FCF3 BD[9AFC0000]        <1>       mov    ebp, m_pix_op_xor_16
11493 0000FCF8 EB05                <1>       jmp    short m_pix_op_xor_w_4
11494                              <1>
11495                              <1>       ; 32 bit true colors
11496                              <1> m_pix_op_xor_w_3:
11497 0000FCFA BD[B7FC0000]        <1>       mov    ebp, m_pix_op_xor_32
11498                              <1> m_pix_op_xor_w_4:
11499 0000FCFF E917FBFFFF          <1>       jmp    m_pix_op_xor_w_x
11500                              <1>
11501                              <1> m_pix_op_not:
11502                              <1>       ; 06/02/2021
11503                              <1>       ; NOT COLOR (MASKED, full screen)
11504                              <1>       ;
11505                              <1>       ; jump from pix_op_not
11506                              <1>       ;
11507                              <1>       ; INPUT:
11508                              <1>       ;   ecx = [v_siz] ; display page pixel count
11509                              <1>       ;   esi = edi = [v_mem] ; LFB start address
11510                              <1>       ;
11511                              <1>       ;   [maskcolor] = mask color (to be excluded)
11512                              <1>       ;
11513                              <1>       ; OUTPUT:
11514                              <1>       ;      [u.r0] will be > 0 if succesful
11515                              <1>
11516                              <1>       ; Full screen
11517                              <1> m_pix_op_not_0:
11518 0000FD04 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11519 0000FD0B 7715                <1>       ja     short m_pix_op_not_1
11520                              <1>       ; 256 colors (8bpp)
11521                              <1>       ;jmp   short m_pix_op_not_8
11522                              <1> m_pix_op_not_8:
11523                              <1>       ; 8 bit colors (256 colors)
11524 0000FD0D AC                  <1>       lodsb
11525 0000FD0E 3A05[9A120300]      <1>       cmp    al, [maskcolor]
11526 0000FD14 7408                <1>       je     short m_pix_op_not_8_1 ; exclude
11527 0000FD16 F617                <1>       not    byte [edi]
11528 0000FD18 FF05[64030300]      <1>       inc    dword [u.r0] ; +1
11529                              <1> m_pix_op_not_8_1:
11530 0000FD1E 47                  <1>       inc    edi
11531 0000FD1F E2EC                <1>       loop   m_pix_op_not_8
11532 0000FD21 C3                  <1>       retn
11533                              <1> m_pix_op_not_1:
```

```
11534 0000FD22 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11535 0000FD29 7746                <1>       ja     short m_pix_op_not_3 ; 32bpp
11536 0000FD2B 7229                <1>       jb     short m_pix_op_not_2 ; 16bpp
11537                              <1>       ; 24 bit true colors
11538                              <1>       ;jmp   short m_pix_op_not_24
11539                              <1> m_pix_op_not_24:
11540                              <1>       ; 24 bit true colors
11541 0000FD2D 66AD                <1>       lodsw
11542 0000FD2F C1E010              <1>       shl    eax, 16
11543 0000FD32 AC                  <1>       lodsb
11544 0000FD33 C1C010              <1>       rol    eax, 16
11545 0000FD36 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
11546 0000FD3C 7412                <1>       je     short m_pix_op_not_24_1 ; exclude
11547 0000FD3E F7D0                <1>       not    eax
11548 0000FD40 668907              <1>       mov    [edi], ax
11549 0000FD43 C1E810              <1>       shr    eax, 16
11550 0000FD46 884702              <1>       mov    [edi+2], al
11551 0000FD49 8305[64030300]03    <1>       add    dword [u.r0], 3 ; +3
11552                              <1> m_pix_op_not_24_1:
11553 0000FD50 83C703              <1>       add    edi, 3 ; +3
11554 0000FD53 E2D8                <1>       loop   m_pix_op_not_24
11555 0000FD55 C3                  <1>       retn
11556                              <1>       ; 65536 colors (16bpp)
11557                              <1> m_pix_op_not_2:
11558                              <1>       ;jmp   short m_pix_op_not_16
11559                              <1> m_pix_op_not_16:
11560                              <1>       ; 16 bit colors (65536 colors)
11561 0000FD56 66AD                <1>       lodsw
11562 0000FD58 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
11563 0000FD5F 740A                <1>       je     short m_pix_op_not_16_1 ; exclude
11564 0000FD61 66F717              <1>       not    word [edi]
11565 0000FD64 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
11566                              <1> m_pix_op_not_16_1:
11567 0000FD6B 83C702              <1>       add    edi, 2 ; +2
11568 0000FD6E E2E6                <1>       loop   m_pix_op_not_16
11569 0000FD70 C3                  <1>       retn
11570                              <1> m_pix_op_not_3:
11571                              <1>       ; 32 bit true colors
11572                              <1>       ;jmp   short m_pix_op_not_32
11573                              <1> m_pix_op_not_32:
11574                              <1>       ; 32 bit true colors
11575 0000FD71 AD                  <1>       lodsd
11576 0000FD72 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
11577 0000FD78 7409                <1>       je     short m_pix_op_not_32_1 ; exclude
11578 0000FD7A F717                <1>       not    dword [edi]
11579 0000FD7C 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
11580                              <1> m_pix_op_not_32_1:
11581 0000FD83 83C704              <1>       add    edi, 4 ; +4
11582 0000FD86 E2E9                <1>       loop   m_pix_op_not_32
11583 0000FD88 C3                  <1>       retn
11584                              <1>
11585                              <1> m_pix_op_not_w:
11586                              <1>       ; 06/02/2021
11587                              <1>       ; NOT COLOR (MASKED, window)
11588                              <1>       ;
11589                              <1>       ; jump from pix_op_not_w
11590                              <1>       ;
11591                              <1>       ; INPUT:
11592                              <1>       ;   ecx = bytes per row (to be applied)
11593                              <1>       ;   edx = screen width in bytes
11594                              <1>       ;   ebx = row count
11595                              <1>       ;
11596                              <1>       ;   [maskcolor] = mask color (to be excluded)
11597                              <1>       ;
11598                              <1>       ; OUTPUT:
11599                              <1>       ;      [u.r0] will be > 0 if succesful
11600                              <1>
11601                              <1>       ; window
11602                              <1>       ;mov   edi, [v_str] ; LFB start address
11603                              <1>       ;mov   esi, edi
11604                              <1>
11605 0000FD89 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11606 0000FD90 7707                <1>       ja     short m_pix_op_not_w_1
11607                              <1>
11608                              <1>       ; 256 colors (8bpp)
11609 0000FD92 BD[0DFD0000]        <1>       mov    ebp, m_pix_op_not_8
11610 0000FD97 EB1E                <1>       jmp    short m_pix_op_not_w_4
11611                              <1>
11612                              <1> m_pix_op_not_w_1:
11613 0000FD99 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11614 0000FDA0 7710                <1>       ja     short m_pix_op_not_w_3 ; 32bpp
11615 0000FDA2 7207                <1>       jb     short m_pix_op_not_w_2 ; 16bpp
11616                              <1>
11617                              <1>       ; 24 bit true colors
11618 0000FDA4 BD[2DFD0000]        <1>       mov    ebp, m_pix_op_not_24
11619 0000FDA9 EB0C                <1>       jmp    short m_pix_op_not_w_4
11620                              <1>
11621                              <1>       ; 65536 colors (16bpp)
11622                              <1> m_pix_op_not_w_2:
11623 0000FDAB BD[56FD0000]        <1>       mov    ebp, m_pix_op_not_16
11624 0000FDB0 EB05                <1>       jmp    short m_pix_op_not_w_4
11625                              <1>
11626                              <1>       ; 32 bit true colors
11627                              <1> m_pix_op_not_w_3:
11628 0000FDB2 BD[71FD0000]        <1>       mov    ebp, m_pix_op_not_32
11629                              <1> m_pix_op_not_w_4:
11630 0000FDB7 E95FFAFFFF          <1>       jmp    m_pix_op_not_w_x
11631                              <1>
11632                              <1> m_pix_op_neg:
11633                              <1>       ; 06/02/2021
11634                              <1>       ; NEGATIVE COLOR (MASKED, full screen)
11635                              <1>       ;
11636                              <1>       ; jump from pix_op_neg
11637                              <1>       ;
11638                              <1>       ; INPUT:
```

```
11639                                  <1>       ;   ecx = [v_siz] ; display page pixel count
11640                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
11641                                  <1>       ;
11642                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11643                                  <1>       ;
11644                                  <1>       ; OUTPUT:
11645                                  <1>       ;      [u.r0] will be > 0 if succesful
11646                                  <1>
11647                                  <1>       ; Full screen
11648                                  <1> m_pix_op_neg_0:
11649 0000FDBC 803D[89120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
11650 0000FDC3 7715                    <1>       ja     short m_pix_op_neg_1
11651                                  <1>       ; 256 colors (8bpp)
11652                                  <1>       ;jmp   short m_pix_op_neg_8
11653                                  <1> m_pix_op_neg_8:
11654                                  <1>       ; 8 bit colors (256 colors)
11655 0000FDC5 AC                      <1>       lodsb
11656 0000FDC6 3A05[9A120300]          <1>       cmp    al, [maskcolor]
11657 0000FDCC 7408                    <1>       je     short m_pix_op_neg_8_1 ; exclude
11658 0000FDCE F61F                    <1>       neg    byte [edi]
11659 0000FDD0 FF05[64030300]          <1>       inc    dword [u.r0] ; +1
11660                                  <1> m_pix_op_neg_8_1:
11661 0000FDD6 47                      <1>       inc    edi
11662 0000FDD7 E2EC                    <1>       loop   m_pix_op_neg_8
11663 0000FDD9 C3                      <1>       retn
11664                                  <1> m_pix_op_neg_1:
11665 0000FDDA 803D[89120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
11666 0000FDE1 7746                    <1>       ja     short m_pix_op_neg_3 ; 32bpp
11667 0000FDE3 7229                    <1>       jb     short m_pix_op_neg_2 ; 16bpp
11668                                  <1>       ; 24 bit true colors
11669                                  <1>       ;jmp   short m_pix_op_neg_24
11670                                  <1> m_pix_op_neg_24:
11671                                  <1>       ; 24 bit true colors
11672 0000FDE5 66AD                    <1>       lodsw
11673 0000FDE7 C1E010                  <1>       shl    eax, 16
11674 0000FDEA AC                      <1>       lodsb
11675 0000FDEB C1C010                  <1>       rol    eax, 16
11676 0000FDEE 3B05[9A120300]          <1>       cmp    eax, [maskcolor]
11677 0000FDF4 7412                    <1>       je     short m_pix_op_neg_24_1 ; exclude
11678 0000FDF6 F7D8                    <1>       neg    eax
11679 0000FDF8 668907                  <1>       mov    [edi], ax
11680 0000FDFB C1E810                  <1>       shr    eax, 16
11681 0000FDFE 884702                  <1>       mov    [edi+2], al
11682 0000FE01 8305[64030300]03        <1>       add    dword [u.r0], 3 ; +3
11683                                  <1> m_pix_op_neg_24_1:
11684 0000FE08 83C703                  <1>       add    edi, 3 ; +3
11685 0000FE0B E2D8                    <1>       loop   m_pix_op_neg_24
11686 0000FE0D C3                      <1>       retn
11687                                  <1>       ; 65536 colors (16bpp)
11688                                  <1> m_pix_op_neg_2:
11689                                  <1>       ;jmp   short m_pix_op_neg_16
11690                                  <1> m_pix_op_neg_16:
11691                                  <1>       ; 16 bit colors (65536 colors)
11692 0000FE0E 66AD                    <1>       lodsw
11693 0000FE10 663B05[9A120300]        <1>       cmp    ax, [maskcolor]
11694 0000FE17 740A                    <1>       je     short m_pix_op_neg_16_1 ; exclude
11695 0000FE19 66F71F                  <1>       neg    word [edi]
11696 0000FE1C 8305[64030300]02        <1>       add    dword [u.r0], 2 ; +2
11697                                  <1> m_pix_op_neg_16_1:
11698 0000FE23 83C702                  <1>       add    edi, 2 ; +2
11699 0000FE26 E2E6                    <1>       loop   m_pix_op_neg_16
11700 0000FE28 C3                      <1>       retn
11701                                  <1> m_pix_op_neg_3:
11702                                  <1>       ; 32 bit true colors
11703                                  <1>       ;jmp   short m_pix_op_neg_32
11704                                  <1> m_pix_op_neg_32:
11705                                  <1>       ; 32 bit true colors
11706 0000FE29 AD                      <1>       lodsd
11707 0000FE2A 3B05[9A120300]          <1>       cmp    eax, [maskcolor]
11708 0000FE30 7409                    <1>       je     short m_pix_op_neg_32_1 ; exclude
11709 0000FE32 F71F                    <1>       neg    dword [edi]
11710 0000FE34 8305[64030300]04        <1>       add    dword [u.r0], 4 ; +4
11711                                  <1> m_pix_op_neg_32_1:
11712 0000FE3B 83C704                  <1>       add    edi, 4 ; +4
11713 0000FE3E E2E9                    <1>       loop   m_pix_op_neg_32
11714 0000FE40 C3                      <1>       retn
11715                                  <1>
11716                                  <1> m_pix_op_neg_w:
11717                                  <1>       ; 06/02/2021
11718                                  <1>       ; NEGATIVE COLOR (MASKED, window)
11719                                  <1>       ;
11720                                  <1>       ; jump from pix_op_neg_w
11721                                  <1>       ;
11722                                  <1>       ; INPUT:
11723                                  <1>       ;   ecx = bytes per row (to be applied)
11724                                  <1>       ;   edx = screen width in bytes
11725                                  <1>       ;   ebx = row count
11726                                  <1>       ;
11727                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11728                                  <1>       ;
11729                                  <1>       ; OUTPUT:
11730                                  <1>       ;      [u.r0] will be > 0 if succesful
11731                                  <1>
11732                                  <1>       ; window
11733                                  <1>       ;mov   edi, [v_str] ; LFB start address
11734                                  <1>       ;mov   esi, edi
11735                                  <1>
11736 0000FE41 803D[89120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
11737 0000FE48 7707                    <1>       ja     short m_pix_op_neg_w_1
11738                                  <1>
11739                                  <1>       ; 256 colors (8bpp)
11740 0000FE4A BD[C5FD0000]            <1>       mov    ebp, m_pix_op_neg_8
11741 0000FE4F EB1E                    <1>       jmp    short m_pix_op_neg_w_4
11742                                  <1>
11743                                  <1> m_pix_op_neg_w_1:
```

```
11744 0000FE51 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11745 0000FE58 7710                <1>       ja     short m_pix_op_neg_w_3 ; 32bpp
11746 0000FE5A 7207                <1>       jb     short m_pix_op_neg_w_2 ; 16bpp
11747                              <1>
11748                              <1>       ; 24 bit true colors
11749 0000FE5C BD[E5FD0000]        <1>       mov    ebp, m_pix_op_neg_24
11750 0000FE61 EB0C                <1>       jmp    short m_pix_op_neg_w_4
11751                              <1>
11752                              <1>       ; 65536 colors (16bpp)
11753                              <1> m_pix_op_neg_w_2:
11754 0000FE63 BD[0EFE0000]        <1>       mov    ebp, m_pix_op_neg_16
11755 0000FE68 EB05                <1>       jmp    short m_pix_op_neg_w_4
11756                              <1>
11757                              <1>       ; 32 bit true colors
11758                              <1> m_pix_op_neg_w_3:
11759 0000FE6A BD[29FE0000]        <1>       mov    ebp, m_pix_op_neg_32
11760                              <1> m_pix_op_neg_w_4:
11761 0000FE6F E9A7F9FFFF          <1>       jmp    m_pix_op_neg_w_x
11762                              <1>
11763                              <1> m_pix_op_inc:
11764                              <1>       ; 06/02/2021
11765                              <1>       ; INCREASE COLOR (MASKED, full screen)
11766                              <1>       ;
11767                              <1>       ; jump from pix_op_inc
11768                              <1>       ;
11769                              <1>       ; INPUT:
11770                              <1>       ;   ecx = [v_siz] ; display page pixel count
11771                              <1>       ;   esi = edi = [v_mem] ; LFB start address
11772                              <1>       ;
11773                              <1>       ;   [maskcolor] = mask color (to be excluded)
11774                              <1>       ;
11775                              <1>       ; OUTPUT:
11776                              <1>       ;      [u.r0] will be > 0 if succesful
11777                              <1>
11778                              <1>       ; Full screen
11779                              <1> m_pix_op_inc_0:
11780 0000FE74 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11781 0000FE7B 7719                <1>       ja     short m_pix_op_inc_1
11782                              <1>       ; 256 colors (8bpp)
11783                              <1>       ;jmp   short m_pix_op_inc_8
11784                              <1> m_pix_op_inc_8:
11785                              <1>       ; 8 bit colors (256 colors)
11786 0000FE7D AC                  <1>       lodsb
11787 0000FE7E 3A05[9A120300]      <1>       cmp    al, [maskcolor]
11788 0000FE84 740C                <1>       je     short m_pix_op_inc_8_1 ; exclude
11789 0000FE86 FE07                <1>       inc    byte [edi]
11790 0000FE88 7502                <1>       jnz    short m_pix_op_inc_8_0
11791 0000FE8A FE0F                <1>       dec    byte [edi]
11792                              <1> m_pix_op_inc_8_0:
11793 0000FE8C FF05[64030300]      <1>       inc    dword [u.r0] ; +1
11794                              <1> m_pix_op_inc_8_1:
11795 0000FE92 47                  <1>       inc    edi
11796 0000FE93 E2E8                <1>       loop   m_pix_op_inc_8
11797 0000FE95 C3                  <1>       retn
11798                              <1> m_pix_op_inc_1:
11799 0000FE96 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11800 0000FE9D 7752                <1>       ja     short m_pix_op_inc_3 ; 32bpp
11801 0000FE9F 7230                <1>       jb     short m_pix_op_inc_2 ; 16bpp
11802                              <1>       ; 24 bit true colors
11803                              <1>       ;jmp   short m_pix_op_inc_24
11804                              <1> m_pix_op_inc_24:
11805                              <1>       ; 24 bit true colors
11806 0000FEA1 66AD                <1>       lodsw
11807 0000FEA3 C1E010              <1>       shl    eax, 16
11808 0000FEA6 AC                  <1>       lodsb
11809 0000FEA7 C1C010              <1>       rol    eax, 16
11810 0000FEAA 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
11811 0000FEB0 7419                <1>       je     short m_pix_op_inc_24_1 ; exclude
11812 0000FEB2 40                  <1>       inc    eax
11813 0000FEB3 3DFFFFFF00          <1>       cmp    eax, 0FFFFFFh
11814 0000FEB8 7601                <1>       jna    short m_pix_op_inc_24_0
11815 0000FEBA 48                  <1>       dec    eax
11816                              <1> m_pix_op_inc_24_0:
11817 0000FEBB 668907              <1>       mov    [edi], ax
11818 0000FEBE C1E810              <1>       shr    eax, 16
11819 0000FEC1 884702              <1>       mov    [edi+2], al
11820 0000FEC4 8305[64030300]03    <1>       add    dword [u.r0], 3 ; +3
11821                              <1> m_pix_op_inc_24_1:
11822 0000FECB 83C703              <1>       add    edi, 3 ; +3
11823 0000FECE E2D1                <1>       loop   m_pix_op_inc_24
11824 0000FED0 C3                  <1>       retn
11825                              <1>       ; 65536 colors (16bpp)
11826                              <1> m_pix_op_inc_2:
11827                              <1>       ;jmp   short m_pix_op_inc_16
11828                              <1> m_pix_op_inc_16:
11829                              <1>       ; 16 bit colors (65536 colors)
11830 0000FED1 66AD                <1>       lodsw
11831 0000FED3 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
11832 0000FEDA 740F                <1>       je     short m_pix_op_inc_16_1 ; exclude
11833 0000FEDC 66FF07              <1>       inc    word [edi]
11834 0000FEDF 7503                <1>       jnz    short m_pix_op_inc_16_0
11835 0000FEE1 66FF0F              <1>       dec    word [edi]
11836                              <1> m_pix_op_inc_16_0:
11837 0000FEE4 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
11838                              <1> m_pix_op_inc_16_1:
11839 0000FEEB 83C702              <1>       add    edi, 2 ; +2
11840 0000FEEE E2E1                <1>       loop   m_pix_op_inc_16
11841 0000FEF0 C3                  <1>       retn
11842                              <1> m_pix_op_inc_3:
11843                              <1>       ; 32 bit true colors
11844                              <1>       ;jmp   short m_pix_op_inc_32
11845                              <1> m_pix_op_inc_32:
11846                              <1>       ; 32 bit true colors
11847 0000FEF1 AD                  <1>       lodsd
11848 0000FEF2 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
```

```
11849 0000FEF8 740D                <1>       je     short m_pix_op_inc_32_1 ; exclude
11850 0000FEFA FF07                <1>       inc    dword [edi]
11851 0000FEFC 7502                <1>       jnz    short m_pix_op_inc_32_0
11852 0000FEFE FF0F                <1>       dec    dword [edi]
11853                              <1> m_pix_op_inc_32_0:
11854 0000FF00 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
11855                              <1> m_pix_op_inc_32_1:
11856 0000FF07 83C704              <1>       add    edi, 4 ; +4
11857 0000FF0A E2E5                <1>       loop   m_pix_op_inc_32
11858 0000FF0C C3                  <1>       retn
11859                              <1>
11860                              <1> m_pix_op_inc_w:
11861                              <1>       ; 06/02/2021
11862                              <1>       ; INCREASE COLOR (MASKED, window)
11863                              <1>       ;
11864                              <1>       ; jump from pix_op_inc_w
11865                              <1>       ;
11866                              <1>       ; INPUT:
11867                              <1>       ;   ecx = bytes per row (to be applied)
11868                              <1>       ;   edx = screen width in bytes
11869                              <1>       ;   ebx = row count
11870                              <1>       ;
11871                              <1>       ;   [maskcolor] = mask color (to be excluded)
11872                              <1>       ;
11873                              <1>       ; OUTPUT:
11874                              <1>       ;      [u.r0] will be > 0 if succesful
11875                              <1>
11876                              <1>       ; window
11877                              <1>       ;mov   edi, [v_str] ; LFB start address
11878                              <1>       ;mov   esi, edi
11879                              <1>
11880 0000FF0D 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11881 0000FF14 7707                <1>       ja     short m_pix_op_inc_w_1
11882                              <1>
11883                              <1>       ; 256 colors (8bpp)
11884 0000FF16 BD[7DFE0000]        <1>       mov    ebp, m_pix_op_inc_8
11885 0000FF1B EB1E                <1>       jmp    short m_pix_op_inc_w_4
11886                              <1>
11887                              <1> m_pix_op_inc_w_1:
11888 0000FF1D 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11889 0000FF24 7710                <1>       ja     short m_pix_op_inc_w_3 ; 32bpp
11890 0000FF26 7207                <1>       jb     short m_pix_op_inc_w_2 ; 16bpp
11891                              <1>
11892                              <1>       ; 24 bit true colors
11893 0000FF28 BD[A1FE0000]        <1>       mov    ebp, m_pix_op_inc_24
11894 0000FF2D EB0C                <1>       jmp    short m_pix_op_inc_w_4
11895                              <1>
11896                              <1>       ; 65536 colors (16bpp)
11897                              <1> m_pix_op_inc_w_2:
11898 0000FF2F BD[D1FE0000]        <1>       mov    ebp, m_pix_op_inc_16
11899 0000FF34 EB05                <1>       jmp    short m_pix_op_inc_w_4
11900                              <1>
11901                              <1>       ; 32 bit true colors
11902                              <1> m_pix_op_inc_w_3:
11903 0000FF36 BD[F1FE0000]        <1>       mov    ebp, m_pix_op_inc_32
11904                              <1> m_pix_op_inc_w_4:
11905 0000FF3B E9DBF8FFFF          <1>       jmp    m_pix_op_inc_w_x
11906                              <1>
11907                              <1> m_pix_op_dec:
11908                              <1>       ; 06/02/2021
11909                              <1>       ; DECREASE COLOR (MASKED, full screen)
11910                              <1>       ;
11911                              <1>       ; jump from pix_op_dec
11912                              <1>       ;
11913                              <1>       ; INPUT:
11914                              <1>       ;   ecx = [v_siz] ; display page pixel count
11915                              <1>       ;   esi = edi = [v_mem] ; LFB start address
11916                              <1>       ;
11917                              <1>       ;   [maskcolor] = mask color (to be excluded)
11918                              <1>       ;
11919                              <1>       ; OUTPUT:
11920                              <1>       ;      [u.r0] will be > 0 if succesful
11921                              <1>
11922                              <1>       ; Full screen
11923                              <1> m_pix_op_dec_0:
11924 0000FF40 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
11925 0000FF47 7719                <1>       ja     short m_pix_op_dec_1
11926                              <1>       ; 256 colors (8bpp)
11927                              <1>       ;jmp   short m_pix_op_dec_8
11928                              <1> m_pix_op_dec_8:
11929                              <1>       ; 8 bit colors (256 colors)
11930 0000FF49 AC                  <1>       lodsb
11931 0000FF4A 3A05[9A120300]      <1>       cmp    al, [maskcolor]
11932 0000FF50 740C                <1>       je     short m_pix_op_dec_8_1 ; exclude
11933 0000FF52 FE0F                <1>       dec    byte [edi]
11934 0000FF54 7902                <1>       jns    short m_pix_op_dec_8_0
11935 0000FF56 FE07                <1>       inc    byte [edi]
11936                              <1> m_pix_op_dec_8_0:
11937 0000FF58 FF05[64030300]      <1>       inc    dword [u.r0] ; +1
11938                              <1> m_pix_op_dec_8_1:
11939 0000FF5E 47                  <1>       inc    edi
11940 0000FF5F E2E8                <1>       loop   m_pix_op_dec_8
11941 0000FF61 C3                  <1>       retn
11942                              <1> m_pix_op_dec_1:
11943 0000FF62 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
11944 0000FF69 774D                <1>       ja     short m_pix_op_dec_3 ; 32bpp
11945 0000FF6B 722B                <1>       jb     short m_pix_op_dec_2 ; 16bpp
11946                              <1>       ; 24 bit true colors
11947                              <1>       ;jmp   short m_pix_op_dec_24
11948                              <1> m_pix_op_dec_24:
11949                              <1>       ; 24 bit true colors
11950 0000FF6D 66AD                <1>       lodsw
11951 0000FF6F C1E010              <1>       shl    eax, 16
11952 0000FF72 AC                  <1>       lodsb
11953 0000FF73 C1C010              <1>       rol    eax, 16
```

```
11954 0000FF76 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
11955 0000FF7C 7414                <1>       je     short m_pix_op_dec_24_1 ; exclude
11956 0000FF7E 48                  <1>       dec    eax
11957 0000FF7F 7901                <1>       jns    short m_pix_op_dec_24_0
11958 0000FF81 40                  <1>       inc    eax
11959                              <1> m_pix_op_dec_24_0:
11960 0000FF82 668907              <1>       mov    [edi], ax
11961 0000FF85 C1E810              <1>       shr    eax, 16
11962 0000FF88 884702              <1>       mov    [edi+2], al
11963 0000FF8B 8305[64030300]03    <1>       add    dword [u.r0], 3 ; +3
11964                              <1> m_pix_op_dec_24_1:
11965 0000FF92 83C703              <1>       add    edi, 3 ; +3
11966 0000FF95 E2D6                <1>       loop   m_pix_op_dec_24
11967 0000FF97 C3                  <1>       retn
11968                              <1>       ; 65536 colors (16bpp)
11969                              <1> m_pix_op_dec_2:
11970                              <1>       ;jmp   short m_pix_op_dec_16
11971                              <1> m_pix_op_dec_16:
11972                              <1>       ; 16 bit colors (65536 colors)
11973 0000FF98 66AD                <1>       lodsw
11974 0000FF9A 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
11975 0000FFA1 740F                <1>       je     short m_pix_op_dec_16_1 ; exclude
11976 0000FFA3 66FF0F              <1>       dec    word [edi]
11977 0000FFA6 7903                <1>       jns    short m_pix_op_dec_16_0
11978 0000FFA8 66FF07              <1>       inc    word [edi]
11979                              <1> m_pix_op_dec_16_0:
11980 0000FFAB 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
11981                              <1> m_pix_op_dec_16_1:
11982 0000FFB2 83C702              <1>       add    edi, 2 ; +2
11983 0000FFB5 E2E1                <1>       loop   m_pix_op_dec_16
11984 0000FFB7 C3                  <1>       retn
11985                              <1> m_pix_op_dec_3:
11986                              <1>       ; 32 bit true colors
11987                              <1>       ;jmp   short m_pix_op_dec_32
11988                              <1> m_pix_op_dec_32:
11989                              <1>       ; 32 bit true colors
11990 0000FFB8 AD                  <1>       lodsd
11991 0000FFB9 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
11992 0000FFBF 740D                <1>       je     short m_pix_op_dec_32_1 ; exclude
11993 0000FFC1 FF0F                <1>       dec    dword [edi]
11994 0000FFC3 7902                <1>       jns    short m_pix_op_dec_32_0
11995 0000FFC5 FF07                <1>       inc    dword [edi]
11996                              <1> m_pix_op_dec_32_0:
11997 0000FFC7 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
11998                              <1> m_pix_op_dec_32_1:
11999 0000FFCE 83C704              <1>       add    edi, 4 ; +4
12000 0000FFD1 E2E5                <1>       loop   m_pix_op_dec_32
12001 0000FFD3 C3                  <1>       retn
12002                              <1>
12003                              <1> m_pix_op_dec_w:
12004                              <1>       ; 06/02/2021
12005                              <1>       ; DECREASE COLOR (MASKED, window)
12006                              <1>       ;
12007                              <1>       ; jump from pix_op_dec_w
12008                              <1>       ;
12009                              <1>       ; INPUT:
12010                              <1>       ;   ecx = bytes per row (to be applied)
12011                              <1>       ;   edx = screen width in bytes
12012                              <1>       ;   ebx = row count
12013                              <1>       ;
12014                              <1>       ;   [maskcolor] = mask color (to be excluded)
12015                              <1>       ;
12016                              <1>       ; OUTPUT:
12017                              <1>       ;      [u.r0] will be > 0 if succesful
12018                              <1>
12019                              <1>       ; window
12020                              <1>       ;mov   edi, [v_str] ; LFB start address
12021                              <1>       ;mov   esi, edi
12022                              <1>
12023 0000FFD4 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
12024 0000FFDB 7707                <1>       ja     short m_pix_op_dec_w_1
12025                              <1>
12026                              <1>       ; 256 colors (8bpp)
12027 0000FFDD BD[49FF0000]        <1>       mov    ebp, m_pix_op_dec_8
12028 0000FFE2 EB1E                <1>       jmp    short m_pix_op_dec_w_4
12029                              <1>
12030                              <1> m_pix_op_dec_w_1:
12031 0000FFE4 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
12032 0000FFEB 7710                <1>       ja     short m_pix_op_dec_w_3 ; 32bpp
12033 0000FFED 7207                <1>       jb     short m_pix_op_dec_w_2 ; 16bpp
12034                              <1>
12035                              <1>       ; 24 bit true colors
12036 0000FFEF BD[6DFF0000]        <1>       mov    ebp, m_pix_op_dec_24
12037 0000FFF4 EB0C                <1>       jmp    short m_pix_op_dec_w_4
12038                              <1>
12039                              <1>       ; 65536 colors (16bpp)
12040                              <1> m_pix_op_dec_w_2:
12041 0000FFF6 BD[98FF0000]        <1>       mov    ebp, m_pix_op_dec_16
12042 0000FFFB EB05                <1>       jmp    short m_pix_op_dec_w_4
12043                              <1>
12044                              <1>       ; 32 bit true colors
12045                              <1> m_pix_op_dec_w_3:
12046 0000FFFD BD[B8FF0000]        <1>       mov    ebp, m_pix_op_dec_32
12047                              <1> m_pix_op_dec_w_4:
12048 00010002 E914F8FFFF          <1>       jmp    m_pix_op_dec_w_x
12049                              <1>
12050                              <1> sysvideo_39:
12051                              <1>       ; 15/02/2021
12052                              <1>       ; 07/02/2021, 08/02/2021
12053                              <1>       ; 03/01/2021, 04/01/2021
12054                              <1>       ; 23/11/2020
12055                              <1>       ; BH = 3
12056                              <1>       ; PIXEL READ/WRITE
12057                              <1>
12058                              <1>       ; 07/02/2021
```

```
12059                                  <1>       ; 04/01/2021 (TRDOS 386 v2.0.3)
12060 00010007 80FB03                  <1>       cmp    bl, 3
12061 0001000A 761A                    <1>       jna    short sysvideo_39_1
12062                                  <1>       ; 07/02/2021
12063 0001000C 80FB06                  <1>       cmp    bl, 6
12064 0001000F 7705                    <1>       ja     short sysvideo_39_0
12065 00010011 E91A010000              <1>       jmp    sysvideo_39_31
12066                                  <1> sysvideo_39_0:
12067                                  <1>       ; error
12068 00010016 B3FF                    <1>       mov    bl, 0FFh
12069 00010018 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12070 0001001E 895D10                  <1>       mov    [ebp+16], ebx ; EBX
12071 00010021 E9A6D7FFFF              <1>       jmp    sysret
12072                                  <1> sysvideo_39_1:
12073 00010026 803D[9A6E0000]FF        <1>       cmp    byte [CRT_MODE], 0FFh
12074 0001002D 7312                    <1>       jnb    short sysvideo_39_2 ; SVGA (VESA VBE) video mode
12075                                  <1>
12076                                  <1>       ; Std VGA or CGA mode
12077 0001002F 81C200000A00            <1>       add    edx, 0A0000h
12078 00010035 72DF                    <1>       jc     short sysvideo_39_0
12079 00010037 81FAFFFF0A00            <1>       cmp    edx, 0AFFFFh
12080 0001003D 77D7                    <1>       ja     short sysvideo_39_0
12081 0001003F EB1E                    <1>       jmp    short sysvideo_39_3 ; 8bpp
12082                                  <1>
12083                                  <1> sysvideo_39_2:
12084                                  <1>       ; use current vbe (svga) video mode
12085                                  <1>
12086                                  <1>       ; get LFB address
12087 00010041 A1[2C120300]            <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
12088 00010046 09C0                    <1>       or     eax, eax
12089 00010048 74CC                    <1>       jz     short sysvideo_39_0
12090 0001004A 3B15[30120300]          <1>       cmp    edx, [LFB_SIZE]
12091 00010050 73C4                    <1>       jnb    short sysvideo_39_0
12092                                  <1>
12093 00010052 01C2                    <1>       add    edx, eax
12094                                  <1>       ;jc    short sysvideo_39_0
12095                                  <1>
12096                                  <1>       ; Pixel read/write in VESA VBE (2/3) video mode
12097                                  <1>       ; Video memory at Linear Frame Buffer base address
12098                                  <1>
12099 00010054 8A3D[38120300]          <1>       mov    bh, [LFB_Info+LFBINFO.bpp]
12100                                  <1>
12101 0001005A 80FF08                  <1>       cmp    bh, 8 ; 8bpp
12102 0001005D 775D                    <1>       ja     short sysvideo_39_17
12103                                  <1>
12104                                  <1>       ; 8 bits per pixel
12105                                  <1> sysvideo_39_3:
12106 0001005F 80FB01                  <1>       cmp    bl, 1 ; 1 = write pixel
12107 00010062 7406                    <1>       je     short sysvideo_39_5
12108 00010064 7712                    <1>       ja     short sysvideo_39_8
12109                                  <1> sysvideo_39_4:
12110                                  <1>       ; read pixel (8bpp)
12111 00010066 8A02                    <1>       mov    al, [edx]
12112                                  <1>       ;mov   [u.r0], al
12113                                  <1>       ;jmp   sysret
12114 00010068 EB04                    <1>       jmp    short sysvideo_39_7
12115                                  <1> sysvideo_39_5:
12116                                  <1>       ; write pixel (8bpp)
12117 0001006A 88C8                    <1>       mov    al, cl
12118                                  <1> sysvideo_39_6:
12119 0001006C 8802                    <1>       mov    [edx], al
12120                                  <1> sysvideo_39_7:
12121 0001006E A2[64030300]            <1>       mov    [u.r0], al
12122 00010073 E954D7FFFF              <1>       jmp    sysret
12123                                  <1> sysvideo_39_8:
12124 00010078 80FB03                  <1>       cmp    bl, 3 ; mix
12125 0001007B 7208                    <1>       jb     short sysvideo_39_9
12126                                  <1>       ; mix  pixel colors (8bpp)
12127 0001007D 8A02                    <1>       mov    al, [edx]
12128 0001007F 00C8                    <1>       add    al, cl
12129 00010081 D0D8                    <1>       rcr    al, 1
12130 00010083 EBE7                    <1>       jmp    short sysvideo_39_6
12131                                  <1> sysvideo_39_9:
12132                                  <1>        ; swap pixel colors (8bpp)
12133 00010085 88C8                    <1>       mov    al, cl
12134 00010087 8602                    <1>       xchg   [edx], al
12135 00010089 EBE3                    <1>       jmp    short sysvideo_39_7
12136                                  <1>
12137                                  <1>       ; 16 bits per pixel
12138                                  <1> sysvideo_39_10:
12139 0001008B 80FB01                  <1>       cmp    bl, 1 ; 1 = write pixel
12140 0001008E 7406                    <1>       je     short sysvideo_39_12
12141 00010090 7714                    <1>       ja     short sysvideo_39_15
12142                                  <1> sysvideo_39_11:
12143                                  <1>       ; read pixel (16bpp)
12144 00010092 8B02                    <1>       mov    eax, [edx]
12145                                  <1>       ;mov   [u.r0], ax
12146                                  <1>       ;jmp   sysret
12147 00010094 EB05                    <1>       jmp    short sysvideo_39_14
12148                                  <1> sysvideo_39_12:
12149                                  <1>       ; write pixel (16bpp)
12150 00010096 89C8                    <1>       mov    eax, ecx
12151                                  <1> sysvideo_39_13:
12152 00010098 668902                  <1>       mov    [edx], ax
12153                                  <1> sysvideo_39_14:
12154 0001009B 66A3[64030300]          <1>       mov    [u.r0], ax
12155 000100A1 E926D7FFFF              <1>       jmp    sysret
12156                                  <1> sysvideo_39_15:
12157 000100A6 80FB03                  <1>       cmp    bl, 3 ; mix
12158 000100A9 720A                    <1>       jb     short sysvideo_39_16
12159                                  <1>       ; mix  pixel colors (16bpp)
12160 000100AB 8B02                    <1>       mov    eax, [edx]
12161 000100AD 6601C8                  <1>       add    ax, cx
12162 000100B0 66D1D8                  <1>       rcr    ax, 1
12163 000100B3 EBE3                    <1>       jmp    short sysvideo_39_13
```

```
12164                                  <1> sysvideo_39_16:
12165                                  <1>         ; swap pixel colors (16bpp)
12166 000100B5 89C8                    <1>         mov     eax, ecx
12167 000100B7 668702                  <1>         xchg    [edx], ax
12168 000100BA EBDF                    <1>         jmp     short sysvideo_39_14
12169                                  <1> sysvideo_39_17:
12170 000100BC 80FF18                  <1>         cmp     bh, 24
12171 000100BF 7743                    <1>         ja      short sysvideo_39_24
12172 000100C1 72C8                    <1>         jb      short sysvideo_39_10
12173                                  <1>
12174                                  <1>         ; 24 bits per pixel
12175 000100C3 81E1FFFFFF00            <1>         and     ecx, 0FFFFFFh
12176 000100C9 80FB01                  <1>         cmp     bl, 1 ; 1 = write pixel
12177 000100CC 7406                    <1>         je      short sysvideo_39_19
12178 000100CE 7712                    <1>         ja      short sysvideo_39_22
12179                                  <1> sysvideo_39_18:
12180                                  <1>         ; read pixel (24bpp)
12181 000100D0 8B02                    <1>         mov     eax, [edx]
12182                                  <1>         ;and    eax, 0FFFFFFh
12183                                  <1>         ;mov    [u.r0], eax
12184                                  <1>         ;jmp    sysret
12185 000100D2 EB04                    <1>         jmp     short sysvideo_39_21
12186                                  <1> sysvideo_39_19:
12187                                  <1>         ; write pixel (24bpp)
12188 000100D4 89C8                    <1>         mov     eax, ecx
12189                                  <1> sysvideo_39_20:
12190                                  <1>         ;and    eax, 0FFFFFFh
12191 000100D6 8902                    <1>         mov     [edx], eax
12192                                  <1> sysvideo_39_21:
12193 000100D8 A3[64030300]            <1>         mov     [u.r0], eax
12194 000100DD E9EAD6FFFF              <1>         jmp     sysret
12195                                  <1> sysvideo_39_22:
12196 000100E2 80FB03                  <1>         cmp     bl, 3 ; mix
12197 000100E5 720D                    <1>         jb      short sysvideo_39_23
12198                                  <1>         ; mix  pixel colors (24bpp)
12199 000100E7 8B02                    <1>         mov     eax, [edx]
12200 000100E9 25FFFFFF00              <1>         and     eax, 0FFFFFFh
12201                                  <1>         ;and    ecx, 0FFFFFFh
12202 000100EE 01C8                    <1>         add     eax, ecx
12203 000100F0 D1D8                    <1>         rcr     eax, 1
12204 000100F2 EBE2                    <1>         jmp     short sysvideo_39_20
12205                                  <1> sysvideo_39_23:
12206                                  <1>         ; swap pixel colors (24bpp)
12207 000100F4 89C8                    <1>         mov     eax, ecx
12208                                  <1>         ;and    eax, 0FFFFFFh
12209 000100F6 668702                  <1>         xchg    [edx], ax
12210 000100F9 C1C810                  <1>         ror     eax, 16
12211 000100FC 884202                  <1>         mov     [edx+2], al
12212 000100FF C1C010                  <1>         rol     eax, 16
12213 00010102 EBD4                    <1>         jmp     short sysvideo_39_21
12214                                  <1>
12215                                  <1>         ; 32 bits per pixel
12216                                  <1> sysvideo_39_24:
12217 00010104 80FB01                  <1>         cmp     bl, 1 ; 1 = write pixel
12218 00010107 7406                    <1>         je      short sysvideo_39_26
12219 00010109 7712                    <1>         ja      short sysvideo_39_29
12220                                  <1> sysvideo_39_25:
12221                                  <1>         ; read pixel (32bpp)
12222 0001010B 8B02                    <1>         mov     eax, [edx]
12223                                  <1>         ;mov    [u.r0], eax
12224                                  <1>         ;jmp    sysret
12225 0001010D EB04                    <1>         jmp     short sysvideo_39_28
12226                                  <1> sysvideo_39_26:
12227                                  <1>         ; write pixel (32bpp)
12228 0001010F 89C8                    <1>         mov     eax, ecx
12229                                  <1> sysvideo_39_27:
12230 00010111 8902                    <1>         mov     [edx], eax
12231                                  <1> sysvideo_39_28:
12232 00010113 A3[64030300]            <1>         mov     [u.r0], eax
12233 00010118 E9AFD6FFFF              <1>         jmp     sysret
12234                                  <1> sysvideo_39_29:
12235 0001011D 80FB03                  <1>         cmp     bl, 3 ; mix
12236 00010120 7208                    <1>         jb      short sysvideo_39_30
12237                                  <1>         ; mix  pixel colors (32bpp)
12238 00010122 8B02                    <1>         mov     eax, [edx]
12239 00010124 01C8                    <1>         add     eax, ecx
12240 00010126 D1D8                    <1>         rcr     eax, 1
12241 00010128 EBE7                    <1>         jmp     short sysvideo_39_27
12242                                  <1> sysvideo_39_30:
12243                                  <1>         ; swap pixel colors (32bpp)
12244 0001012A 89C8                    <1>         mov     eax, ecx
12245 0001012C 8702                    <1>         xchg    [edx], eax
12246 0001012E EBE3                    <1>         jmp     short sysvideo_39_28
12247                                  <1>
12248                                  <1> sysvideo_39_31:
12249                                  <1>         ; 06/03/2021
12250                                  <1>         ; 08/02/2021
12251                                  <1>         ; 07/02/2021
12252                                  <1>         ; BL = 4 -> read pixels from user defined positions
12253                                  <1>         ; BL = 5 -> write single color pixels to user defined pos.
12254                                  <1>         ; BL = 6 -> write multi color pixels to user defined pos.
12255                                  <1>         ; ECX = color (CL, CX, ECX)
12256                                  <1>         ; EDX = number of pixels
12257                                  <1>         ; ESI = user buffer contains dword pixel positions
12258                                  <1>         ;       (and dword colors for BL input = 6)
12259                                  <1>         ; EDI = user's pixel color buff (destination) for BL = 4
12260                                  <1>
12261 00010130 890D[9A120300]          <1>         mov     [maskcolor], ecx
12262 00010136 89D5                    <1>         mov     ebp, edx ; number of pixels
12263 00010138 803D[9A6E0000]FF        <1>         cmp     byte [CRT_MODE], 0FFh ; SVGA flag
12264 0001013F 7317                    <1>         jnb     short sysvideo_39_33 ; SVGA (VESA VBE mode)
12265                                  <1>         ; Standard VGA mode
12266 00010141 B900000100              <1>         mov     ecx, 65536 ; Video page size (maximum)
12267 00010146 39CA                    <1>         cmp     edx, ecx
12268 00010148 7709                    <1>         ja      short sysvideo_39_32 ; abnormal value !
```

```
12269 0001014A B800000A00          <1>       mov   eax, 0A0000h ; Video page start address
12270 0001014F B708                <1>       mov   bh, 8  ; 8 bits per pixel (256 colors)
12271 00010151 EB35                <1>       jmp   short sysvideo_39_34
12272                              <1> sysvideo_39_32:
12273                              <1>       ; nonsense! (edx has abnormal value)
12274 00010153 E974D6FFFF          <1>       jmp   sysret
12275                              <1> sysvideo_39_33:
12276                              <1>       ; 06/03/2021
12277 00010158 8A3D[38120300]      <1>       mov   bh, [LFB_Info+LFBINFO.bpp]
12278 0001015E 80FF08              <1>       cmp   bh, 8
12279 00010161 7412                <1>       je    short sysvideo_39_81 ; 8bpp
12280 00010163 89D0                <1>       mov   eax, edx
12281 00010165 80FF10              <1>       cmp   bh, 16
12282 00010168 7409                <1>       je    short sysvideo_39_80 ; 16bpp
12283 0001016A D1E2                <1>       shl   edx, 1
12284 0001016C 80FF20              <1>       cmp   bh, 32
12285 0001016F 7502                <1>       jne   short sysvideo_39_80 ; 24bpp
12286 00010171 D1E0                <1>       shl   eax, 1
12287                              <1> sysvideo_39_80:
12288 00010173 01C2                <1>       add   edx, eax
12289                              <1>       ; edx = number of bytes
12290                              <1> sysvideo_39_81:
12291                              <1>       ; get LFB address
12292 00010175 A1[2C120300]        <1>       mov   eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
12293 0001017A 09C0                <1>       or    eax, eax
12294 0001017C 74D5                <1>       jz    short sysvideo_39_32 ; LFB is not ready !
12295 0001017E 8B0D[30120300]      <1>       mov   ecx, [LFB_SIZE]
12296 00010184 39CA                <1>       cmp   edx, ecx
12297 00010186 77CB                <1>       ja    short sysvideo_39_32 ; abnormal value !
12298                              <1>
12299                              <1>       ; 02/03/2021
12300                              <1>       ; 08/02/2021
12301                              <1>       ;mov   ebp, edx ; pixel count
12302                              <1>       ;shl   ebp, 2 ; byte count (pixel pos: 4 bytes)
12303                              <1>
12304                              <1>       ; 06/03/2021
12305                              <1>       ; bits per pixel (pixel color size)
12306                              <1>       ;mov   bh, [LFB_Info+LFBINFO.bpp]
12307                              <1> sysvideo_39_34:
12308 00010188 C1E502              <1>       shl   ebp, 2 ; 15/02/2021 (byte count)
12309 0001018B A3[8A120300]        <1>       mov   [v_mem], eax ; Save video page start address
12310 00010190 88DE                <1>       mov   dh, bl ; sub function
12311                              <1>       ; 06/03/2021
12312 00010192 883D[89120300]      <1>       mov   [v_bpp], bh ; bits per pixel (color size)
12313                              <1>       ;mov   ebx, [LFB_SIZE]
12314 00010198 89CB                <1>       mov   ebx, ecx ; [LFB_SIZE]
12315                              <1>
12316 0001019A B900080000          <1>       mov   ecx, 2048
12317 0001019F 39CD                <1>       cmp   ebp, ecx
12318 000101A1 7302                <1>       jnb   short sysvideo_39_35
12319 000101A3 89E9                <1>       mov   ecx, ebp ; fix to requested byte count
12320                              <1> sysvideo_39_35:
12321 000101A5 80FE04              <1>       cmp   dh, 4 ; 08/02/2021
12322                              <1>       ;cmp   bl, 4 ; read pixels from user defined positions
12323 000101A8 7605                <1>       jna   short sysvideo_39_36
12324 000101AA E9B2000000          <1>       jmp   sysvideo_39_52
12325                              <1>       ; 08/02/2021
12326                              <1>       ;mov   [buffer8], edi  ; user's destination buff addr
12327                              <1> sysvideo_39_36:
12328                              <1>       ; 08/02/2021
12329                              <1>       ; read pixel positions
12330                              <1>       ; as defined in user's source buffer
12331 000101AF 893D[A2120300]      <1>       mov   [buffer8], edi  ; user's destination buff addr
12332 000101B5 BF00760900          <1>       mov   edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12333                              <1>                                       ; 2028 byte data
12334                              <1>       ; esi = user's source buffer for pixel positions
12335                              <1>       ; ecx = byte count
12336 000101BA E8AF180000          <1>       call  transfer_from_user_buffer
12337 000101BF 7292                <1>       jc    short sysvideo_39_32 ; error
12338                              <1>       ; ecx  = transfer count (bytes)
12339                              <1>
12340 000101C1 57                  <1>       push  edi ; *
12341 000101C2 56                  <1>       push  esi ; **
12342 000101C3 51                  <1>       push  ecx ; ***
12343                              <1>
12344 000101C4 89FE                <1>       mov   esi, edi ; kernel buffer
12345 000101C6 8B15[8A120300]      <1>       mov   edx, [v_mem] ; video memory
12346 000101CC C1E902              <1>       shr   ecx, 2 ; pixel count (within buffer capacity)
12347                              <1>
12348 000101CF 803D[89120300]08    <1>       cmp   byte [v_bpp], 8 ; 8bpp
12349 000101D6 7753                <1>       ja    short sysvideo_39_49
12350                              <1> sysvideo_39_37:
12351                              <1>       ; 8bpp
12352 000101D8 AD                  <1>       lodsd
12353 000101D9 39D8                <1>       cmp   eax, ebx ; < [LFB_SIZE]
12354 000101DB 7309                <1>       jnb   short sysvideo_39_39
12355 000101DD 0FB60402            <1>       movzx eax, byte [edx+eax]
12356                              <1> sysvideo_39_38:
12357 000101E1 AB                  <1>       stosd
12358 000101E2 E2F4                <1>       loop  sysvideo_39_37
12359 000101E4 EB49                <1>       jmp   short sysvideo_39_50
12360                              <1> sysvideo_39_39:
12361                              <1>       ; write black color for improper positions
12362 000101E6 31C0                <1>       xor   eax, eax
12363 000101E8 EBF7                <1>       jmp   short sysvideo_39_38
12364                              <1> sysvideo_39_40:
12365 000101EA 803D[89120300]18    <1>       cmp   byte [v_bpp], 24 ; 24bpp
12366 000101F1 772A                <1>       ja    short sysvideo_39_47 ; 32bpp
12367 000101F3 7216                <1>       jb    short sysvideo_39_44 ; 16bpp
12368                              <1> sysvideo_39_41:
12369                              <1>       ; 24bpp
12370 000101F5 AD                  <1>       lodsd
12371 000101F6 39D8                <1>       cmp   eax, ebx ; < [LFB_SIZE]
12372 000101F8 730D                <1>       jnb   short sysvideo_39_43
12373 000101FA 8B0402              <1>       mov   eax, [edx+eax]
```

```
12374 000101FD 25FFFFFF00          <1>       and    eax, 0FFFFFFh
12375                              <1> sysvideo_39_42:
12376 00010202 AB                  <1>       stosd
12377 00010203 E2F0                <1>       loop   sysvideo_39_41
12378 00010205 EB28                <1>       jmp    short sysvideo_39_50
12379                              <1> sysvideo_39_43:
12380                              <1>       ; write black color for improper positions
12381 00010207 31C0                <1>       xor    eax, eax
12382 00010209 EBF7                <1>       jmp    short sysvideo_39_42
12383                              <1> sysvideo_39_44:
12384                              <1>       ; 16bpp
12385 0001020B AD                  <1>       lodsd
12386 0001020C 39D8                <1>       cmp    eax, ebx ; < [LFB_SIZE]
12387 0001020E 7309                <1>       jnb    short sysvideo_39_46
12388 00010210 0FB70402            <1>       movzx  eax, word [edx+eax]
12389                              <1> sysvideo_39_45:
12390 00010214 AB                  <1>       stosd
12391 00010215 E2F4                <1>       loop   sysvideo_39_44
12392 00010217 EB16                <1>       jmp    short sysvideo_39_50
12393                              <1> sysvideo_39_46:
12394                              <1>       ; write black color for improper positions
12395 00010219 31C0                <1>       xor    eax, eax
12396 0001021B EBF7                <1>       jmp    short sysvideo_39_45
12397                              <1> sysvideo_39_47:
12398                              <1>       ; 32bpp
12399 0001021D AD                  <1>       lodsd
12400 0001021E 39D8                <1>       cmp    eax, ebx ; < [LFB_SIZE]
12401 00010220 7309                <1>       jnb    short sysvideo_39_49
12402 00010222 0FB70402            <1>       movzx  eax, word [edx+eax]
12403                              <1> sysvideo_39_48:
12404 00010226 AB                  <1>       stosd
12405 00010227 E2F4                <1>       loop   sysvideo_39_47
12406 00010229 EB04                <1>       jmp    short sysvideo_39_50
12407                              <1> sysvideo_39_49:
12408                              <1>       ; write black color for improper positions
12409 0001022B 31C0                <1>       xor    eax, eax
12410 0001022D EBF7                <1>       jmp    short sysvideo_39_48
12411                              <1> sysvideo_39_50:
12412 0001022F 59                  <1>       pop    ecx ; transfer count in bytes
12413 00010230 5E                  <1>       pop    esi ; ** ; kernel buffer
12414                              <1>       ;mov   esi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12415                              <1>                                  ; 2048 byte data
12416 00010231 8B3D[A2120300]      <1>       mov    edi, [buffer8]
12417                              <1>       ; edi = user's destination buffer for pixel colors
12418                              <1>       ; ecx = byte count
12419 00010237 E8E8170000          <1>       call   transfer_to_user_buffer
12420 0001023C 5E                  <1>       pop    esi ; *
12421 0001023D 7266                <1>       jc     short sysvideo_39_56 ; error
12422                              <1>       ; ecx  = transfer count (bytes)
12423 0001023F 89C8                <1>       mov    eax, ecx
12424 00010241 C1E802              <1>       shr    eax, 2
12425 00010244 0105[64030300]      <1>       add    [u.r0], eax ; transfer count (in pixels)
12426                              <1>
12427 0001024A 29CD                <1>       sub    ebp, ecx
12428 0001024C 7657                <1>       jna    short sysvideo_39_56 ; completed/finished
12429 0001024E 01CE                <1>       add    esi, ecx ; next position in source buffer
12430                              <1>       ;add   [buffer8], ecx ; next pos in destination buff
12431 00010250 01CF                <1>       add    edi, ecx
12432 00010252 66B90008            <1>       mov    cx, 2048 ; new count, limit: kernel buff size
12433 00010256 39CD                <1>       cmp    ebp, ecx ; remain >= limit ?
12434 00010258 7302                <1>       jnb    short sysvideo_39_51 ; yes
12435 0001025A 89E9                <1>       mov    ecx, ebp ; fix byte count to remain bytes
12436                              <1> sysvideo_39_51:
12437 0001025C E94EFFFFFF          <1>       jmp    sysvideo_39_36
12438                              <1>
12439                              <1> sysvideo_39_52:
12440 00010261 80FE05              <1>       cmp    dh, 5 ; 08/02/2021
12441                              <1>       ;cmp   bl, 5 ; write pixels to user defined positions
12442 00010264 7605                <1>       jna    short sysvideo_39_53
12443 00010266 E9A1000000          <1>       jmp    sysvideo_39_66
12444                              <1> sysvideo_39_53:
12445                              <1>       ; single color pixel writing
12446 0001026B BF00760900          <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12447                              <1>                                  ; 2028 byte data
12448                              <1>       ; esi = user's source buffer for pixel positions
12449                              <1>       ; ecx = byte count
12450 00010270 E8F9170000          <1>       call   transfer_from_user_buffer
12451 00010275 722E                <1>       jc     short sysvideo_39_56 ; error
12452                              <1>       ; ecx = transfer count (bytes)
12453                              <1>
12454                              <1>       ; write pixels by using (user) defined positions
12455                              <1>       ; ecx = byte count (1,2,3,4 times pixel count)
12456                              <1>       ; edi = system buffer address
12457                              <1>
12458 00010277 56                  <1>       push   esi ; *
12459 00010278 51                  <1>       push   ecx ; **
12460                              <1>
12461 00010279 89FE                <1>       mov    esi, edi
12462 0001027B 8B3D[8A120300]      <1>       mov    edi, [v_mem]
12463                              <1>
12464                              <1>       ; 08/02/2021
12465 00010281 C1E902              <1>       shr    ecx, 2 ; pixel count
12466 00010284 8B15[9A120300]      <1>       mov    edx, [maskcolor]
12467                              <1>       ;mov   ebx, [v_siz]
12468                              <1>
12469 0001028A 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
12470 00010291 7717                <1>       ja     short sysvideo_39_57
12471                              <1> sysvideo_39_54:
12472                              <1>       ; 8bpp
12473 00010293 AD                  <1>       lodsd
12474 00010294 39D8                <1>       cmp    eax, ebx ; < [v_siz]
12475 00010296 7309                <1>       jnb    short sysvideo_39_55
12476 00010298 881407              <1>       mov    [edi+eax], dl
12477                              <1>       ; 06/03/2021
12478 0001029B FF05[64030300]      <1>       inc    dword [u.r0]
```

```
12479                                  <1> sysvideo_39_55:
12480 000102A1 E2F0                    <1>       loop   sysvideo_39_54
12481 000102A3 EB50                    <1>       jmp    short sysvideo_39_64
12482                                  <1> sysvideo_39_56:
12483 000102A5 E922D5FFFF              <1>       jmp    sysret
12484                                  <1> sysvideo_39_57:
12485 000102AA 803D[89120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
12486 000102B1 7732                    <1>       ja     short sysvideo_39_62 ; 32bpp
12487 000102B3 721D                    <1>       jb     short sysvideo_39_60 ; 16bpp
12488                                  <1> sysvideo_39_58:
12489                                  <1>       ; 24bpp
12490 000102B5 AD                      <1>       lodsd
12491 000102B6 39D8                    <1>       cmp    eax, ebx ; < [v_siz]
12492 000102B8 7314                    <1>       jnb    short sysvideo_39_59
12493 000102BA 881407                  <1>       mov    [edi+eax], dl
12494 000102BD 40                      <1>       inc    eax
12495 000102BE C1CA08                  <1>       ror    edx, 8
12496 000102C1 66891407                <1>       mov    [edi+eax], dx
12497 000102C5 C1C208                  <1>       rol    edx, 8
12498 000102C8 FF05[64030300]          <1>       inc    dword [u.r0]
12499                                  <1> sysvideo_39_59:
12500 000102CE E2E5                    <1>       loop   sysvideo_39_58
12501 000102D0 EB23                    <1>       jmp    short sysvideo_39_64
12502                                  <1> sysvideo_39_60:
12503                                  <1>       ; 16bpp
12504 000102D2 AD                      <1>       lodsd
12505 000102D3 39D8                    <1>       cmp    eax, ebx ; < [v_siz]
12506 000102D5 730A                    <1>       jnb    short sysvideo_39_61
12507 000102D7 66891407                <1>       mov    [edi+eax], dx
12508 000102DB FF05[64030300]          <1>       inc    dword [u.r0]
12509                                  <1> sysvideo_39_61:
12510 000102E1 E2EF                    <1>       loop   sysvideo_39_60
12511 000102E3 EB10                    <1>       jmp    short sysvideo_39_64
12512                                  <1> sysvideo_39_62:
12513                                  <1>       ; 32bpp
12514 000102E5 AD                      <1>       lodsd
12515 000102E6 39D8                    <1>       cmp    eax, ebx ; < [v_siz]
12516 000102E8 7309                    <1>       jnb    short sysvideo_39_63
12517 000102EA 891407                  <1>       mov    [edi+eax], edx
12518 000102ED FF05[64030300]          <1>       inc    dword [u.r0]
12519                                  <1> sysvideo_39_63:
12520 000102F3 E2F0                    <1>       loop   sysvideo_39_62
12521                                  <1> sysvideo_39_64:
12522 000102F5 59                      <1>       pop    ecx ; **
12523 000102F6 5E                      <1>       pop    esi ; *
12524 000102F7 29CD                    <1>       sub    ebp, ecx
12525 000102F9 76AA                    <1>       jna    short sysvideo_39_56
12526 000102FB 01CE                    <1>       add    esi, ecx
12527 000102FD 66B90008                <1>       mov    cx, 2048
12528 00010301 39CD                    <1>       cmp    ebp, ecx
12529 00010303 7302                    <1>       jnb    short sysvideo_39_65
12530 00010305 89E9                    <1>       mov    ecx, ebp
12531                                  <1> sysvideo_39_65:
12532 00010307 E95FFFFFFF              <1>       jmp    sysvideo_39_53
12533                                  <1>
12534                                  <1> sysvideo_39_66:
12535                                  <1>       ; 15/02/2021
12536 0001030C D1E5                    <1>       shl    ebp, 1 ; 8 bytes per pixel (position&color)
12537                                  <1> sysvideo_39_67:
12538 0001030E 66B90008                <1>       mov    cx, 2048
12539 00010312 39CD                    <1>       cmp    ebp, ecx
12540 00010314 7302                    <1>       jnb    short sysvideo_39_68
12541 00010316 89E9                    <1>       mov    ecx, ebp
12542                                  <1> sysvideo_39_68:
12543                                  <1>       ; multi colors pixel writing
12544 00010318 BF00760900              <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12545                                  <1>                                      ; 2048 byte data
12546                                  <1>       ; esi = user's source buffer for pixel positions
12547                                  <1>       ; ecx = byte count
12548 0001031D E84C170000              <1>       call   transfer_from_user_buffer
12549 00010322 7281                    <1>       jc     short sysvideo_39_56 ; error
12550                                  <1>       ; ecx  = transfer count
12551                                  <1>
12552                                  <1>       ; write pixels & colors as defined in user buffer
12553                                  <1>       ; ecx = byte count (2,4,6,8 times pixel count)
12554                                  <1>       ; edi = system buffer address
12555                                  <1>
12556 00010324 56                      <1>       push   esi ; **
12557 00010325 51                      <1>       push   ecx ; *
12558                                  <1>
12559 00010326 89FE                    <1>       mov    esi, edi
12560 00010328 8B3D[8A120300]          <1>       mov    edi, [v_mem]
12561                                  <1>
12562                                  <1>       ; 08/02/2021
12563 0001032E C1E903                  <1>       shr    ecx, 3 ; pixel count
12564                                  <1>
12565                                  <1>       ;mov   ebx, [v_siz]
12566                                  <1>
12567 00010331 803D[89120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
12568 00010338 7715                    <1>       ja     short sysvideo_39_71
12569                                  <1> sysvideo_39_69:
12570                                  <1>       ; 8bpp
12571 0001033A AD                      <1>       lodsd  ; position
12572 0001033B 89C2                    <1>       mov    edx, eax
12573 0001033D AD                      <1>       lodsd  ; color
12574 0001033E 39DA                    <1>       cmp    edx, ebx ; < [v_siz]
12575 00010340 7309                    <1>       jnb    short sysvideo_39_70
12576 00010342 880417                  <1>       mov    [edi+edx], al
12577                                  <1>       ; 06/03/2021
12578 00010345 FF05[64030300]          <1>       inc    dword [u.r0]
12579                                  <1> sysvideo_39_70:
12580 0001034B E2ED                    <1>       loop   sysvideo_39_69
12581 0001034D EB51                    <1>       jmp    short sysvideo_39_78
12582                                  <1> sysvideo_39_71:
12583 0001034F 803D[89120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
```

```
12584 00010356 7735              <1>       ja     short sysvideo_39_76 ; 32bpp
12585 00010358 721D              <1>       jb     short sysvideo_39_74 ; 16bpp
12586                            <1> sysvideo_39_72:
12587                            <1>       ; 24bpp
12588 0001035A AD                <1>       lodsd  ; position
12589 0001035B 89C2              <1>       mov    edx, eax
12590 0001035D AD                <1>       lodsd  ; color
12591 0001035E 39DA              <1>       cmp    edx, ebx ; < [v_siz]
12592 00010360 7311              <1>       jnb    short sysvideo_39_73
12593 00010362 880417            <1>       mov    [edi+edx], al
12594 00010365 42                <1>       inc    edx
12595 00010366 C1E808            <1>       shr    eax, 8
12596 00010369 66890417          <1>       mov    [edi+edx], ax
12597 0001036D FF05[64030300]    <1>       inc    dword [u.r0]
12598                            <1> sysvideo_39_73:
12599 00010373 E2E5              <1>       loop   sysvideo_39_72
12600 00010375 EB29              <1>       jmp    short sysvideo_39_78
12601                            <1> sysvideo_39_74:
12602                            <1>       ; 16bpp
12603 00010377 AD                <1>       lodsd  ; position
12604 00010378 89C2              <1>       mov    edx, eax
12605 0001037A AD                <1>       lodsd  ; color
12606 0001037B 39DA              <1>       cmp    edx, ebx ; < [v_siz]
12607 0001037D 730A              <1>       jnb    short sysvideo_39_75
12608 0001037F 66890417          <1>       mov    [edi+edx], ax
12609 00010383 FF05[64030300]    <1>       inc    dword [u.r0]
12610                            <1> sysvideo_39_75:
12611 00010389 E2EC              <1>       loop   sysvideo_39_74
12612 0001038B EB13              <1>       jmp    short sysvideo_39_78
12613                            <1> sysvideo_39_76:
12614                            <1>       ; 32bpp
12615 0001038D AD                <1>       lodsd  ; position
12616 0001038E 89C2              <1>       mov    edx, eax
12617 00010390 AD                <1>       lodsd  ; color
12618 00010391 39DA              <1>       cmp    edx, ebx ; < [v_siz]
12619 00010393 7309              <1>       jnb    short sysvideo_39_77
12620 00010395 890417            <1>       mov    [edi+edx], eax
12621 00010398 FF05[64030300]    <1>       inc    dword [u.r0]
12622                            <1> sysvideo_39_77:
12623 0001039E E2ED              <1>       loop   sysvideo_39_76
12624                            <1> sysvideo_39_78:
12625 000103A0 59                <1>       pop    ecx ; *
12626 000103A1 5E                <1>       pop    esi ; **
12627                            <1>
12628 000103A2 29CD              <1>       sub    ebp, ecx
12629 000103A4 762A              <1>       jna    short sysvideo_39_79
12630 000103A6 01CE              <1>       add    esi, ecx
12631 000103A8 E961FFFFFF        <1>       jmp    sysvideo_39_67
12632                            <1> ;sysvideo_39_79:
12633                            <1> ;     jmp    sysret
12634                            <1>
12635                            <1> sysvideo_16:
12636                            <1>       ; 06/03/2021
12637                            <1>       ; 23/11/2020
12638 000103AD 80FF04            <1>       cmp    bh, 4
12639 000103B0 0F8251FCFFFF      <1>       jb     sysvideo_39 ; bh = 3, pixel r/w
12640 000103B6 771D              <1>       ja     short sysvideo_17
12641                            <1>
12642                            <1>       ; BH = 4
12643                            <1>       ; Direct User Access for CGA video memory.
12644                            <1>       ; Setup user's page tables for direct access to 0B8000h.
12645                            <1>       ;
12646                            <1>       ; Permission checks are not implemented yet !
12647                            <1>       ; (11/07/2016)
12648                            <1>
12649 000103B8 B800800B00        <1>       mov    eax, 0B8000h
12650 000103BD B908000000        <1>       mov    ecx, 8 ; 8 pages (8*4K=32K)
12651 000103C2 89C3              <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
12652 000103C4 E80562FFFF        <1>       call   direct_memory_access
12653                            <1>       ;jc    sysret
12654 000103C9 7205              <1>       jc     short sysvideo_39_79 ; 06/03/2021
12655                            <1>       ; eax = 0B8000h if there is not an error
12656 000103CB A3[64030300]      <1>       mov    [u.r0], eax
12657                            <1> sysvideo_39_79: ; 08/01/2021
12658 000103D0 E9F7D3FFFF        <1>       jmp    sysret
12659                            <1>
12660                            <1> sysvideo_17:
12661                            <1>       ; 23/12/2020
12662                            <1>       ; 11/12/2020
12663                            <1>       ; 10/12/2020
12664                            <1>       ; 23/11/2020
12665 000103D5 80FF06            <1>       cmp    bh, 6
12666 000103D8 740B              <1>       je     short sysvideo_17_0
12667 000103DA 0F82B9000000      <1>       jb     sysvideo_18
12668 000103E0 E913010000        <1>       jmp    sysvideo_20 ; ja
12669                            <1>
12670                            <1> sysvideo_17_0:
12671                            <1>       ; BH = 6
12672                            <1>       ; Direct User Access to Linear Frame Buffer.
12673                            <1>       ; Setup user's page tables for direct access to LFB.
12674                            <1>       ;
12675                            <1>       ; Permission checks are not implemented yet !
12676                            <1>       ; (10/12/2020)
12677                            <1>
12678 000103E5 80FBFF            <1>       cmp    bl, 0FFh ; current video mode
12679 000103E8 722C              <1>       jb     short sysvideo_17_2 ; for desired video mode
12680                            <1>
12681 000103EA 381D[9A6E0000]    <1>       cmp    [CRT_MODE], bl ; VESA VBE video mode ?
12682 000103F0 750E              <1>       jne    short sysvideo_17_1
12683 000103F2 668B0D[1E120300]  <1>       mov    cx, [video_mode]
12684 000103F9 6681E1FF01        <1>       and    cx, 1FFh
12685 000103FE EB29              <1>       jmp    short sysvideo_17_3
12686                            <1> sysvideo_17_1:
12687                            <1>       ; 11/12/2020
12688 00010400 88DF              <1>       mov    bh, bl ; 0FFh
```

```
12689 00010402 8A1D[9A6E0000]      <1>       mov    bl, [CRT_MODE] ; VGA/CGA video mode
12690 00010408 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12691                              <1>       ; 23/12/2020
12692 0001040E 895D10              <1>       mov    [ebp+16], ebx ; return to user with EBX value
12693 00010411 E9B6D3FFFF          <1>       jmp    sysret ; return to user with EAX = 0
12694                              <1> sysvideo_17_2:
12695                              <1>       ; bl = VESA video mode - 100h
12696 00010416 B701                <1>       mov    bh, 1 ; bx = 1XXh
12697 00010418 53                  <1>       push   ebx ; requested vesa video mode
12698 00010419 E8F036FFFF          <1>       call   vbe_biosfn_return_current_mode
12699 0001041E 59                  <1>       pop    ecx ; requested vesa video mode
12700 0001041F 6681E3FF01          <1>       and    bx, 1FFh
12701 00010424 6639D9              <1>       cmp    cx, bx
12702 00010427 7564                <1>       jne    short sysvideo_17_8
12703                              <1> sysvideo_17_3:
12704 00010429 663B0D[2A120300]    <1>       cmp    cx, [LFB_Info+LFBINFO.mode]
12705 00010430 755B                <1>       jne    short sysvideo_17_8
12706                              <1> sysvideo_17_4:
12707                              <1>       ; 11/12/2020
12708 00010432 A1[2C120300]        <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
12709                              <1>       ; 21/12/2020
12710 00010437 09C0                <1>       or     eax, eax
12711 00010439 744D                <1>       jz     short sysvideo_17_7
12712                              <1>       ;
12713 0001043B 8B0D[30120300]      <1>       mov    ecx, [LFB_Info+LFBINFO.LFB_size] ; buff size
12714 00010441 89C3                <1>       mov    ebx, eax ; user's address = physical address
12715                              <1>       ;push  ebx
12716 00010443 51                  <1>       push   ecx
12717                              <1>       ; 21/12/2020
12718 00010444 81C1FF0F0000        <1>       add    ecx, 4095  ; PAGESIZE - 1
12719                              <1>       ; 14/12/2020
12720 0001044A C1E90C              <1>       shr    ecx, 12  ; convert bytes to pages
12721 0001044D E87C61FFFF          <1>       call   direct_memory_access
12722 00010452 5A                  <1>       pop    edx  ; linear frame buffer size in bytes
12723                              <1>       ;pop   eax  ; linear frame buffer address (physical)
12724 00010453 7233                <1>       jc     short sysvideo_17_7 ; [u.r0] = eax = 0
12725                              <1> sysvideo_17_5:
12726 00010455 668B0D[36120300]    <1>       mov    cx, [LFB_Info+LFBINFO.Y_res] ; screen height
12727 0001045C C1E110              <1>       shl    ecx, 16
12728 0001045F 668B0D[34120300]    <1>       mov    cx,  [LFB_Info+LFBINFO.X_res] ; screen width
12729 00010466 31DB                <1>       xor    ebx, ebx
12730 00010468 8A1D[38120300]      <1>       mov    bl, [LFB_Info+LFBINFO.bpp] ; bits per pixel
12731 0001046E 8A3D[2A120300]      <1>       mov    bh, [LFB_Info+LFBINFO.mode] ; XX part of 1XXh
12732                              <1> sysvideo_26_4: ; 23/12/2020
12733 00010474 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12734 0001047A 895514              <1>       mov    [ebp+20], edx ; return to user with EDX value
12735 0001047D 895D10              <1>       mov    [ebp+16], ebx ; EBX
12736 00010480 894D18              <1>       mov    [ebp+24], ecx ; ECX
12737                              <1> sysvideo_17_6:
12738 00010483 A3[64030300]        <1>       mov    [u.r0], eax ; LFB address
12739                              <1> sysvideo_17_7:
12740 00010488 E93FD3FFFF          <1>       jmp    sysret
12741                              <1> sysvideo_17_8:
12742                              <1>       ; cx = mode
12743                              <1>       ; 21/12/2020
12744 0001048D 80CD40              <1>       or     ch, 40h  ; Linear frame buffer flag
12745 00010490 E89B34FFFF          <1>       call   _vbe_biosfn_return_mode_info
12746 00010495 72F1                <1>       jc     short sysvideo_17_7
12747 00010497 EB99                <1>       jmp    short sysvideo_17_4
12748                              <1>
12749                              <1> sysvideo_18:
12750                              <1>       ; BH = 5
12751                              <1>       ; Direct User Access for VGA video memory.
12752                              <1>       ; Setup user's page tables for direct access to 0A0000h.
12753                              <1>       ;
12754                              <1>       ; Permission checks are not implemented yet !
12755                              <1>       ; (11/07/2016)
12756                              <1>
12757 00010499 B800000A00          <1>       mov    eax, 0A0000h
12758 0001049E B910000000          <1>       mov    ecx, 16 ; 16 pages (16*4K=64K)
12759 000104A3 89C3                <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
12760 000104A5 E82461FFFF          <1>       call   direct_memory_access
12761 000104AA 0F821CD3FFFF        <1>       jc     sysret
12762                              <1>       ; eax = 0A0000h if there is not an error
12763 000104B0 A3[64030300]        <1>       mov    [u.r0], eax
12764 000104B5 E912D3FFFF          <1>       jmp    sysret
12765                              <1>
12766                              <1> sysvideo_19:
12767                              <1>       ; 22/01/2021
12768                              <1>       ; 12/12/2020
12769                              <1>       ; 11/12/2020
12770                              <1>       ; 23/11/2020
12771                              <1>       ; BH = 7
12772                              <1>       ; Get (Super/Extended VGA) mode
12773                              <1>       ; and Linear Frame Buffer info.
12774                              <1>
12775                              <1>       ; 22/01/2021
12776 000104BA B3FF                <1>       mov    bl, 0FFh
12777                              <1>       ; 11/12/2020
12778                              <1>       ;cmp   byte [CRT_MODE], 0FFh ; (extended mode?)
12779                              <1>       ; 22/01/2021
12780 000104BC 381D[9A6E0000]      <1>       cmp    [CRT_MODE], bl  ; 0FFh
12781                              <1>       ;jb    sysvideo_17_1; not a VESA VBE mode
12782                              <1>       ; 12/12/2020
12783 000104C2 7305                <1>       jnb    short sysvideo_19_0
12784 000104C4 E937FFFFFF          <1>       jmp    sysvideo_17_1
12785                              <1>
12786                              <1> sysvideo_19_0:
12787 000104C9 E84036FFFF          <1>       call   vbe_biosfn_return_current_mode
12788 000104CE 6681E3FF01          <1>       and    bx, 1FFh
12789 000104D3 663B1D[2A120300]    <1>       cmp    bx, [LFB_Info+LFBINFO.mode]
12790 000104DA 7510                <1>       jne    short sysvideo_19_2
12791                              <1> sysvideo_19_1:
12792 000104DC A1[2C120300]        <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
12793 000104E1 8B15[30120300]      <1>       mov    edx, [LFB_Info+LFBINFO.LFB_size]
```

```
12794 000104E7 E969FFFFFF          <1>       jmp    sysvideo_17_5
12795                              <1> sysvideo_19_2:
12796 000104EC E83F34FFFF          <1>       call   _vbe_biosfn_return_mode_info
12797 000104F1 73E9                <1>       jnc    short sysvideo_19_1
12798 000104F3 E9D4D2FFFF          <1>       jmp    sysret
12799                              <1>
12800                              <1> sysvideo_20:
12801                              <1>       ; 11/12/2020
12802                              <1>       ; 23/11/2020
12803 000104F8 80FF08              <1>       cmp    bh, 8
12804 000104FB 72BD                <1>       jb     short sysvideo_19 ; video mode & lfb info
12805 000104FD 0F8780000000        <1>       ja     sysvideo_21 ; 12/12/2020
12806                              <1>
12807                              <1>       ; BH = 8
12808                              <1>       ; Set (Super/Extended VGA) mode & return LFB info
12809                              <1>       ;
12810                              <1>
12811                              <1>       ; 11/12/2020
12812 00010503 80FBFF              <1>       cmp    bl, 0FFh  ; CGA/VGA mode ?
12813 00010506 7318                <1>       jnb    short sysvideo_20_1
12814                              <1>
12815                              <1>       ;xor   ah, ah
12816 00010508 88D8                <1>       mov    al, bl
12817                              <1> sysvideo_20_0:
12818 0001050A E8F411FFFF          <1>       call   _int10h  ; uses vbe3 pmi32 option
12819 0001050F 83F8FF              <1>       cmp    eax, 0FFFFFFFFh ; -1
12820 00010512 7459                <1>       je     short sysvideo_20_3 ; error
12821                              <1>
12822                              <1>       ; 11/12/2020
12823                              <1>       ; alternative (it does not use vbe3 pmi32)
12824                              <1>       ;push  eax
12825                              <1>       ;call  _set_mode
12826                              <1>       ;pop   eax
12827                              <1>       ;jc    short sysvideo_20_3
12828                              <1>
12829                              <1>       ;inc   eax
12830 00010514 FEC0                <1>       inc    al
12831                              <1>       ;mov   [u.r0], ax ; video mode + 1
12832 00010516 A2[64030300]        <1>       mov    [u.r0], al
12833 0001051B E9ACD2FFFF          <1>       jmp    sysret
12834                              <1>
12835                              <1> sysvideo_20_1:
12836                              <1>       ; cx = vesa video mode
12837 00010520 6689C8              <1>       mov    ax, cx
12838 00010523 663D0001            <1>       cmp    ax, 100h
12839 00010527 72E1                <1>       jb     short sysvideo_20_0 ; VGA/CGA mode
12840 00010529 663DFF01            <1>       cmp    ax, 1FFh
12841                              <1>       ;ja    short sysvideo_20_4 ; not valid
12842 0001052D 773E                <1>       ja     short sysvideo_20_3
12843 0001052F 50                  <1>       push   eax
12844 00010530 6689C3              <1>       mov    bx, ax
12845 00010533 66B8024F            <1>       mov    ax, 4F02h
12846                              <1>
12847                              <1>       ; simulate _int10h (int 31h) for func 4F02h
12848                              <1>       ;pushfd
12849                              <1>       ;push  cs
12850                              <1>       ;push  sysvideo_20_1_retn
12851                              <1>       ;push  es ; *
12852                              <1>       ;push  ds ; **      ; SAVE WORK AND PARAMETER REGISTERS
12853                              <1>       ;jmp   VBE_func
12854                              <1> ;sysvideo_20_1_retn:
12855                              <1>
12856 00010537 E8C711FFFF          <1>       call   _int10h ; simulate int 10h (int 31h)
12857                              <1>
12858 0001053C 6683F84F            <1>       cmp    ax, 004Fh
12859 00010540 58                  <1>       pop    eax
12860 00010541 752A                <1>       jne    short sysvideo_20_3 ; error
12861                              <1>       ;pop   eax
12862 00010543 40                  <1>       inc    eax
12863 00010544 A3[64030300]        <1>       mov    [u.r0], eax ; video mode + 1
12864 00010549 09D2                <1>       or     edx, edx ; is LFBINFO requested by user ?
12865                              <1>       ;jz    short sysvideo_20_4
12866 0001054B 7420                <1>       jz     short sysvideo_20_3 ; no
12867                              <1>
12868                              <1>       ; 11/12/2020
12869                              <1>       ; Check LFBINFO table/structure
12870                              <1>       ; (it is set by vbe2 'vbe_biosfn_set_mode'
12871                              <1>       ; but if vbe3 vbios pmi is in use,
12872                              <1>       ; it will not set LFBINFO table)
12873                              <1>
12874 0001054D 52                  <1>       push   edx
12875 0001054E 48                  <1>       dec    eax  ; video mode
12876 0001054F BE[2A120300]        <1>       mov    esi, LFB_Info
12877 00010554 663B06              <1>       cmp    ax, [esi+LFBINFO.mode]
12878 00010557 7407                <1>       je     short sysvideo_20_2
12879                              <1>
12880 00010559 E8D233FFFF          <1>       call   _vbe_biosfn_return_mode_info
12881                              <1>       ;jnc   short sysvideo_20_2
12882 0001055E 7212                <1>       jc     short sysvideo_20_4 ; edx = 0
12883                              <1>
12884                              <1>       ;; clear LFBINFO table for invalidating
12885                              <1>       ;mov   ecx, LFBINFO.size ; 16
12886                              <1>       ;mov   edi, esi ; LFB_Info table address
12887                              <1>       ;xor   eax, eax
12888                              <1>       ;rep   stosb
12889                              <1>
12890                              <1> sysvideo_20_2:
12891                              <1>       ;pop   ecx
12892                              <1>       ;mov   edi, ecx ; user buffer
12893 00010560 5F                  <1>       pop    edi
12894 00010561 B910000000          <1>       mov    ecx, LFBINFO.size ; 16
12895 00010566 E8B9140000          <1>       call   transfer_to_user_buffer ; fast transfer
12896 0001056B 7206                <1>       jc     short sysvideo_20_5
12897                              <1>
12898                              <1>       ;jmp   sysret
```

```
12899                                  <1> sysvideo_20_3:
12900                                  <1>       ;pop   eax ; [u.r0] = 0
12901                                  <1> ;sysvideo_20_4:
12902 0001056D E95AD2FFFF              <1>       jmp    sysret
12903                                  <1>
12904                                  <1> sysvideo_20_4:
12905 00010572 5A                      <1>       pop    edx
12906                                  <1> sysvideo_20_5:
12907 00010573 31D2                    <1>       xor    edx, edx ; 0
12908                                  <1>       ; edx = 0 -> invalid LFBINFO data
12909 00010575 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12910 0001057B 895514                  <1>       mov    [ebp+20], edx ; return to user with EDX value
12911 0001057E E949D2FFFF              <1>       jmp    sysret
12912                                  <1>
12913                                  <1> sysvideo_21:
12914                                  <1>       ; 04/01/2021
12915                                  <1>       ; 03/12/2020
12916 00010583 80FF0A                  <1>       cmp    bh, 10
12917 00010586 0F82AA010000            <1>       jb     sysvideo_22 ; VESA VBE3 pmi parms
12918                                  <1>       ; 23/12/2020
12919 0001058C 0F845F020000            <1>       je     sysvideo_26 ; Video memory mapping
12920                                  <1>
12921                                  <1>       ; 04/01/2020
12922 00010592 80FF0B                  <1>       cmp    bh, 11
12923 00010595 0F87E1020000            <1>       ja     sysvideo_27
12924                                  <1>
12925                                  <1>       ; BH = 11
12926                                  <1>       ; set/read DAC color registers (for 8bpp)
12927                                  <1>
12928 0001059B 80FB04                  <1>       cmp    bl, 4
12929 0001059E 0F83AB000000            <1>       jnb    sysvideo_21_7; BMP file type palette
12930                                  <1>                           ; handling
12931 000105A4 F6C301                  <1>       test   bl, 1
12932 000105A7 7555                    <1>       jnz    short sysvideo_21_4 ; set/write DAC colors
12933                                  <1>
12934                                  <1>       ; Read DAC color register or all DAC color registers
12935 000105A9 F6C302                  <1>       test   bl, 2  ; read single DAC color register
12936 000105AC 7424                    <1>       jz     short sysvideo_21_2 ; read all DAC color regs
12937                                  <1>
12938                                  <1>       ; read single DAC color register
12939                                  <1>       ; CL = DAC color register (index)
12940                                  <1>
12941 000105AE 66BAC703                <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
12942 000105B2 88C8                    <1>       mov    al, cl ; DAC color register
12943 000105B4 31C9                    <1>       xor    ecx, ecx ; (this may not be necessary)
12944 000105B6 EE                      <1>       out    dx, al
12945                                  <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
12946 000105B7 B2C9                    <1>       mov    dl, 0C9h
12947 000105B9 EC                      <1>       in     al, dx
12948 000105BA 88C4                    <1>       mov    ah, al ; red
12949 000105BC EC                      <1>       in     al, dx
12950 000105BD 88C1                    <1>       mov    cl, al ; green
12951 000105BF EC                      <1>       in     al, dx
12952 000105C0 88C5                    <1>       mov    ch, al ; blue
12953 000105C2 C1E108                  <1>       shl    ecx, 8
12954 000105C5 88E1                    <1>       mov    cl, ah ; red
12955                                  <1>       ; CL = Red, CH = Green, byte 3 = Blue, byte 4 = 0
12956                                  <1> sysvideo_21_0:
12957 000105C7 890D[64030300]          <1>       mov    [u.r0], ecx
12958                                  <1> sysvideo_21_1:
12959 000105CD E9FAD1FFFF              <1>       jmp    sysret
12960                                  <1> sysvideo_21_2:
12961                                  <1>       ; read all DAC color registers
12962 000105D2 89CB                    <1>       mov    ebx, ecx ; user's buffer address
12963 000105D4 BF00600900              <1>       mov    edi, VBE3STACKADDR
12964 000105D9 89FE                    <1>       mov    esi, edi
12965 000105DB B900030000              <1>       mov    ecx, 768 ; 256*3
12966 000105E0 51                      <1>       push   ecx
12967 000105E1 66BAC703                <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
12968 000105E5 28C0                    <1>       sub    al, al ; 0
12969 000105E7 EE                      <1>       out    dx, al
12970                                  <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
12971 000105E8 B2C9                    <1>       mov    dl, 0C9h
12972                                  <1> sysvideo_21_3:
12973 000105EA EC                      <1>       in     al, dx
12974 000105EB AA                      <1>       stosb
12975 000105EC EC                      <1>       in     al, dx
12976 000105ED AA                      <1>       stosb
12977 000105EE EC                      <1>       in     al, dx
12978 000105EF AA                      <1>       stosb
12979 000105F0 E2F8                    <1>       loop   sysvideo_21_3
12980 000105F2 59                      <1>       pop    ecx
12981                                  <1>
12982 000105F3 89DF                    <1>       mov    edi, ebx ; user's buffer address
12983                                  <1>       ;mov   esi, VBE3STACKADDR
12984                                  <1>       ;mov   ecx, 256*3 = 768
12985 000105F5 E82A140000              <1>       call   transfer_to_user_buffer
12986 000105FA 72D1                    <1>       jc     short sysvideo_21_1
12987                                  <1>       ;mov   [u.r0], ecx ; actual transfer count
12988 000105FC EBC9                    <1>       jmp    short sysvideo_21_0
12989                                  <1>
12990                                  <1> sysvideo_21_4:
12991                                  <1>       ; Set/Write DAC color register or all registers
12992 000105FE F6C302                  <1>       test   bl, 2 ; write/set single DAC color register
12993 00010601 741C                    <1>       jz     short sysvideo_21_5 ; set all DAC color regs
12994                                  <1>
12995                                  <1>       ; set single DAC color register
12996                                  <1>       ; CL = DAC color register (index)
12997                                  <1>       ; (byte 1 = Red, byte 2 = Green, byte 3 = Blue)
12998                                  <1>
12999 00010603 66BAC803                <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13000 00010607 89C8                    <1>       mov    eax, ecx ; DAC color register (index)
13001 00010609 C1E910                  <1>       shr    ecx, 16  ; CL = green, AH = Red
13002 0001060C EE                      <1>       out    dx, al
13003                                  <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
```

```
13004 0001060D FEC2               <1>       inc    dl
13005 0001060F 88E0               <1>       mov    al, ah ; Red
13006 00010611 EE                 <1>       out    dx, al
13007 00010612 88C8               <1>       mov    al, cl ; Green
13008 00010614 EE                 <1>       out    dx, al
13009 00010615 88E8               <1>       mov    al, ch ; Blue
13010 00010617 EE                 <1>       out    dx, al
13011                             <1>       ;rol   ecx, 8
13012 00010618 C1E108             <1>       shl    ecx, 8 ; 21/02/2021
13013 0001061B 88E1               <1>       mov    cl, ah ; Red
13014                             <1>              ; ecx = 00BBGGRRh
13015 0001061D EBA8               <1>       jmp    short sysvideo_21_0
13016                             <1>
13017                             <1> sysvideo_21_5:
13018                             <1>       ; write/set all DAC color registers
13019 0001061F 89CE               <1>       mov    esi, ecx ; user's buffer address
13020 00010621 BF00600900         <1>       mov    edi, VBE3STACKADDR
13021 00010626 89FB               <1>       mov    ebx, edi
13022 00010628 B900030000         <1>       mov    ecx, 768 ; 256*3
13023 0001062D E83C140000         <1>       call   transfer_from_user_buffer
13024 00010632 7299               <1>       jc     short sysvideo_21_1
13025 00010634 890D[64030300]     <1>       mov    [u.r0], ecx ; actual transfer count
13026                             <1>
13027 0001063A 89DE               <1>       mov    esi, ebx ; VBE3STACKADDR
13028 0001063C 66BAC803           <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13029 00010640 28C0               <1>       sub    al, al ; 0
13030 00010642 EE                 <1>       out    dx, al
13031                             <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13032 00010643 FEC2               <1>       inc    dl
13033                             <1> sysvideo_21_6:
13034 00010645 AC                 <1>       lodsb
13035 00010646 EE                 <1>       out    dx, al
13036 00010647 AC                 <1>       lodsb
13037 00010648 EE                 <1>       out    dx, al
13038 00010649 AC                 <1>       lodsb
13039 0001064A EE                 <1>       out    dx, al
13040 0001064B E2F8               <1>       loop   sysvideo_21_6
13041 0001064D EB30               <1>       jmp    short sysvideo_21_9
13042                             <1>
13043                             <1> sysvideo_21_7:
13044                             <1>       ; BMP file type palette handling
13045                             <1>
13046 0001064F F6C301             <1>       test   bl, 1
13047 00010652 756E               <1>       jnz    short sysvideo_21_12 ; set/write DAC colors
13048                             <1>
13049                             <1>       ; Read DAC color register or all DAC color registers
13050 00010654 F6C302             <1>       test   bl, 2  ; read single DAC color register
13051 00010657 742B               <1>       jz     short sysvideo_21_10 ; read all DAC color regs
13052                             <1>
13053                             <1>       ; read single DAC color register
13054                             <1>       ; CL = DAC color register (index)
13055                             <1>
13056 00010659 66BAC703           <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
13057 0001065D 88C8               <1>       mov    al, cl ; DAC color register
13058 0001065F 31C9               <1>       xor    ecx, ecx
13059 00010661 EE                 <1>       out    dx, al
13060                             <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13061 00010662 B2C9               <1>       mov    dl, 0C9h
13062 00010664 EC                 <1>       in     al, dx
13063 00010665 C0E002             <1>       shl    al, 2
13064 00010668 88C5               <1>       mov    ch, al ; red
13065 0001066A EC                 <1>       in     al, dx
13066 0001066B C0E002             <1>       shl    al, 2
13067 0001066E 88C1               <1>       mov    cl, al ; green
13068 00010670 EC                 <1>       in     al, dx
13069 00010671 C0E002             <1>       shl    al, 2
13070                             <1>       ; 21/02/2021
13071 00010674 C1E108             <1>       shl    ecx, 8
13072 00010677 88C1               <1>       mov    cl, al ; blue
13073                             <1>       ; CL = Blue, CH = Green, byte 3 = Red, byte 4 = 0
13074                             <1> sysvideo_21_8:
13075 00010679 890D[64030300]     <1>       mov    [u.r0], ecx
13076                             <1> sysvideo_21_9:
13077 0001067F E948D1FFFF         <1>       jmp    sysret
13078                             <1> sysvideo_21_10:
13079                             <1>       ; read all DAC color registers
13080 00010684 89CD               <1>       mov    ebp, ecx ; user's buffer address
13081 00010686 BF00600900         <1>       mov    edi, VBE3STACKADDR
13082 0001068B 89FE               <1>       mov    esi, edi
13083 0001068D B900040000         <1>       mov    ecx, 1024 ; 256*4
13084 00010692 51                 <1>       push   ecx
13085 00010693 66BAC703           <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
13086 00010697 28C0               <1>       sub    al, al ; 0
13087 00010699 EE                 <1>       out    dx, al
13088                             <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13089 0001069A B2C9               <1>       mov    dl, 0C9h
13090                             <1> sysvideo_21_11:
13091 0001069C 31DB               <1>       xor    ebx, ebx
13092 0001069E EC                 <1>       in     al, dx ; Red
13093 0001069F C0E002             <1>       shl    al, 2
13094 000106A2 88C7               <1>       mov    bh, al
13095 000106A4 EC                 <1>       in     al, dx ; Green
13096 000106A5 C0E002             <1>       shl    al, 2
13097 000106A8 88C3               <1>       mov    bl, al
13098 000106AA EC                 <1>       in     al, dx ; Blue
13099 000106AB C0E002             <1>       shl    al, 2
13100 000106AE C1E308             <1>       shl    ebx, 8
13101 000106B1 89D8               <1>       mov    eax, ebx ; 00RRGGBBh
13102 000106B3 AB                 <1>       stosd
13103 000106B4 E2E6               <1>       loop   sysvideo_21_11
13104 000106B6 59                 <1>       pop    ecx
13105                             <1>
13106 000106B7 89EF               <1>       mov    edi, ebp ; user's buffer address
13107                             <1>       ;mov   esi, VBE3STACKADDR
13108                             <1>       ;mov   ecx, 1024 = 4*256
```

```
13109 000106B9 E866130000          <1>       call   transfer_to_user_buffer
13110 000106BE 72BF                <1>       jc     short sysvideo_21_9
13111                              <1>       ;mov   [u.r0], ecx ; actual transfer count
13112 000106C0 EBB7                <1>       jmp    short sysvideo_21_8
13113                              <1>
13114                              <1> sysvideo_21_12:
13115                              <1>       ; Set/Write DAC color register or all registers
13116 000106C2 F6C302              <1>       test   bl, 2 ; write/set single DAC color register
13117 000106C5 7427                <1>       jz     short sysvideo_21_13 ; set all DAC color regs
13118                              <1>
13119                              <1>       ; set single DAC color register
13120                              <1>       ; CL = DAC color register (index)
13121                              <1>       ; (byte 1 = Blue, byte 2 = Green, byte 3 = Red)
13122                              <1>
13123 000106C7 66BAC803           <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13124 000106CB 88C8                <1>       mov    al, cl ; DAC color register (index)
13125 000106CD 88EC                <1>       mov    ah, ch ; Blue
13126 000106CF C1E910              <1>       shr    ecx, 16
13127 000106D2 EE                  <1>       out    dx, al
13128                              <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13129 000106D3 FEC2                <1>       inc    dl
13130 000106D5 88E8                <1>       mov    al, ch ; Red
13131 000106D7 C0E802              <1>       shr    al, 2
13132 000106DA EE                  <1>       out    dx, al
13133 000106DB 88C8                <1>       mov    al, cl ; Green
13134 000106DD C0E802              <1>       shr    al, 2
13135 000106E0 EE                  <1>       out    dx, al
13136 000106E1 88E0                <1>       mov    al, ah ; Blue
13137 000106E3 C0E802              <1>       shr    al, 2
13138 000106E6 EE                  <1>       out    dx, al
13139                              <1>       ;rol   ecx, 8
13140 000106E7 C1E108              <1>       shl    ecx, 8 ; 21/02/2021
13141 000106EA 88E1                <1>       mov    cl, ah
13142 000106EC EB8B                <1>       jmp    short sysvideo_21_8
13143                              <1>
13144                              <1> sysvideo_21_13:
13145                              <1>       ; write/set all DAC color registers
13146 000106EE 89CE                <1>       mov    esi, ecx ; user's buffer address
13147 000106F0 BF00600900          <1>       mov    edi, VBE3STACKADDR
13148 000106F5 89FB                <1>       mov    ebx, edi
13149 000106F7 B900040000          <1>       mov    ecx, 1024 ; 256*4
13150 000106FC E86D130000          <1>       call   transfer_from_user_buffer
13151 00010701 722E                <1>       jc     short sysvideo_21_15
13152 00010703 890D[64030300]      <1>       mov    [u.r0], ecx ; actual transfer count
13153                              <1>
13154 00010709 89DE                <1>       mov    esi, ebx ; VBE3STACKADDR
13155 0001070B 66BAC803            <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13156 0001070F 28C0                <1>       sub    al, al ; 0
13157 00010711 EE                  <1>       out    dx, al
13158                              <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13159 00010712 FEC2                <1>       inc    dl
13160                              <1> sysvideo_21_14:
13161 00010714 AD                  <1>       lodsd
13162                              <1>       ; byte 0 = Blue, byte 1 = Green, byte 2 = Red
13163                              <1>       ; 21/02/2021
13164 00010715 89C3                <1>       mov    ebx, eax ; BL = Blue, BH = Green
13165 00010717 C1CB08              <1>       ror    ebx, 8 ; BL = Green, BH = Red
13166 0001071A 88F8                <1>       mov    al, bh
13167 0001071C C0E802              <1>       shr    al, 2
13168 0001071F EE                  <1>       out    dx, al ; Red
13169 00010720 88D8                <1>       mov    al, bl
13170 00010722 C0E802              <1>       shr    al, 2
13171 00010725 EE                  <1>       out    dx, al ; Green
13172 00010726 C1C308              <1>       rol    ebx, 8 ; BL = Blue
13173 00010729 88D8                <1>       mov    al, bl
13174 0001072B C0E802              <1>       shr    al, 2
13175 0001072E EE                  <1>       out    dx, al ; Blue
13176 0001072F E2E3                <1>       loop   sysvideo_21_14
13177                              <1> sysvideo_21_15:
13178 00010731 E996D0FFFF          <1>       jmp    sysret
13179                              <1>
13180                              <1> sysvideo_22:
13181                              <1>       ; 28/02/2021
13182                              <1>       ; 22/01/2021
13183                              <1>       ; 17/01/2021
13184                              <1>       ; 04/12/2020
13185                              <1>       ; 03/12/2020
13186                              <1>       ; BH = 9
13187                              <1>       ; Set/Get VESA VBE3 protected mode interface params
13188                              <1>
13189                              <1>       ; 22/01/2021
13190                              <1>       ;cmp   byte [vbe3], 3
13191                              <1>       ;jne   short sysvideo_25 ; not applicable if
13192                              <1>                          ; vbe3 compatible video bios
13193                              <1>                          ; is not detected by kernel
13194 00010736 80FB02              <1>       cmp    bl, 2
13195                              <1>       ;ja    short sysvideo_25 ; bl > 2 not implemented
13196                              <1>       ; 17/01/2021
13197 00010739 7716                <1>       ja     short sysvideo_22_0 ; srvs flag sub function
13198                              <1>       ;jb    short sysvideo_23
13199                              <1>
13200                              <1>       ; 21/01/2021
13201 0001073B 803D[52090000]03    <1>       cmp    byte [vbe3], 3
13202                              <1>       ;jne   short sysvideo_25 ; not applicable if
13203                              <1>                          ; vbe3 compatible video bios
13204                              <1>                          ; is not detected by kernel
13205 00010742 75ED                <1>       jne    short sysvideo_21_15 ; 28/02/2021
13206                              <1>
13207 00010744 80FB01              <1>       cmp    bl, 1
13208 00010747 7673                <1>       jna    short sysvideo_23
13209                              <1>
13210 00010749 8A1D[1C120300]      <1>       mov    bl, [pmi32] ; Video bios 32 bit PMI functions
13211 0001074F EB78                <1>       jmp    short sysvideo_24
13212                              <1>
13213                              <1> sysvideo_22_0:
```

```
13214                                  <1>       ; 17/01/2021
13215                                  <1>       ; save/restore video state user permission
13216 00010751 80FB05                  <1>       cmp    bl, 5
13217 00010754 771E                    <1>       ja     short sysvideo_22_2
13218 00010756 7208                    <1>       jb     short sysvideo_22_1
13219                                  <1>       ; get srvs flag value/status
13220 00010758 8A1D[80120300]          <1>       mov    bl, [srvsf] ; 0 = disabled, 1 = enabled
13221 0001075E EB2C                    <1>       jmp    short sysvideo_22_3
13222                                  <1>
13223                                  <1> sysvideo_22_1:
13224                                  <1>       ; permission (root and multi tasking) check
13225 00010760 E836000000              <1>       call   sysvideo_22_4
13226 00010765 736A                    <1>       jnc    short sysvideo_25 ; not permitted !
13227                                  <1>       ; cf = 1
13228 00010767 80EB03                  <1>       sub    bl, 3 ; disable = 0, enable = 1
13229                                  <1>       ; 22/01/2021
13230 0001076A 881D[80120300]          <1>       mov    [srvsf], bl
13231 00010770 FEC3                    <1>       inc    bl ; 1 = disabled, 2 = enabled
13232 00010772 EB18                    <1>       jmp    short sysvideo_22_3
13233                                  <1>
13234                                  <1> sysvideo_22_2:
13235 00010774 80FB06                  <1>       cmp    bl, 6
13236                                  <1>       ;ja    short sysvideo_25 ; invalid/unimplemented
13237                                  <1>       ; 28/02/2021
13238 00010777 7733                    <1>       ja     short sysvideo_22_6
13239                                  <1>       ; get VESA VBE number/status
13240 00010779 8A25[52090000]          <1>       mov    ah, [vbe3] ; vbe3 = 3, vbe2 = 2, others = 0
13241 0001077F A0[53090000]            <1>       mov    al, [vbe2bios] ; bochs/qemu/vbox emulator status
13242 00010784 66A3[64030300]          <1>       mov    [u.r0], ax
13243 0001078A EB45                    <1>       jmp    short sysvideo_25
13244                                  <1>
13245                                  <1> sysvideo_22_3:
13246                                  <1>       ; 22/01/2021
13247 0001078C 8A3D[81120300]          <1>       mov    bh, [srvso] ; state options (> 80h -> svga)
13248 00010792 66891D[64030300]        <1>       mov    [u.r0], bx ; function result is return value
13249 00010799 EB36                    <1>       jmp    short sysvideo_25
13250                                  <1>
13251                                  <1> sysvideo_22_4:
13252                                  <1>       ; 17/01/2021 - permission will be given by root only
13253 0001079B 803D[82950100]00        <1>       cmp    byte [multi_tasking], 0 ; in single user mode
13254 000107A2 7707                    <1>       ja     short sysvideo_22_5
13255                                  <1>       ; 19/01/2021
13256 000107A4 803D[B0030300]01        <1>       cmp    byte [u.uid], 1 ; ([u.uid] = 0 -> root)
13257                                  <1> sysvideo_22_5:
13258                                  <1>       ; [multi_tasking] = 0 & [u.uid] = 0 -> CF = 1
13259                                  <1>       ; otherwise -> CF = 0
13260 000107AB C3                      <1>       retn
13261                                  <1>
13262                                  <1> sysvideo_22_6:
13263                                  <1>       ; 28/02/2021
13264 000107AC 80FB09                  <1>       cmp    bl, 9
13265 000107AF 7720                    <1>       ja     short sysvideo_25 ; invalid/unimplemented
13266 000107B1 7436                    <1>       je     short sysvideo_22_9
13267 000107B3 80FB08                  <1>       cmp    bl, 8
13268 000107B6 721E                    <1>       jb     short sysvideo_22_7
13269                                  <1>
13270                                  <1>       ; BL = 8
13271                                  <1>       ; Set default true color bpp to 24
13272                                  <1>
13273 000107B8 B318                    <1>       mov    bl, 24
13274                                  <1>       ;mov   [truecolor], al     ; 24bpp (RRGGBBh)
13275                                  <1>       ;mov   [u.r0], al
13276                                  <1>       ;jmp   short sysvideo_25
13277 000107BA EB25                    <1>       jmp    short sysvideo_22_8
13278                                  <1>
13279                                  <1> sysvideo_23:
13280                                  <1>       ; 17/01/2021
13281                                  <1>       ; permission (root and multi tasking) check
13282 000107BC E8DAFFFFFF              <1>       call   sysvideo_22_4
13283 000107C1 730E                    <1>       jnc    short sysvideo_25 ; not permitted !
13284                                  <1>
13285 000107C3 881D[1C120300]          <1>       mov    [pmi32], bl ; 1 = enabled, 0 = disabled
13286                                  <1> sysvideo_24:
13287 000107C9 FEC3                    <1>       inc    bl
13288                                  <1> sysvideo_22_10:    ; 28/02/2021
13289 000107CB 881D[64030300]          <1>       mov    [u.r0], bl ; function result is return value
13290                                  <1> sysvideo_25:
13291 000107D1 E9F6CFFFFF              <1>       jmp    sysret
13292                                  <1>
13293                                  <1> sysvideo_22_7:
13294                                  <1>       ; BL = 7
13295                                  <1>       ; Set default true color bpp to 32
13296                                  <1>       ; (it will set if [VBE3]=3)
13297                                  <1>
13298                                  <1>       ; Note: This sub function is used to set 24bpp
13299                                  <1>       ; VESA VBE video modes to 32bpp.. because,
13300                                  <1>       ; old hardware uses 24 bpp but new video hardware
13301                                  <1>       ; uses 32bpp for same VESA VBE truecolor modes.
13302                                  <1>       ; (For example: VBE mode 112h is 640*480, 24bpp but
13303                                  <1>       ; new hardware uses/apply it as 640*480, 32bpp.)
13304                                  <1>       ; So, TRDOS 386 v2.0.3 kernel will check [truecolor]
13305                                  <1>       ; status is 32 bpp or not and it will change 24bpp
13306                                  <1>       ; to 32bpp if default [truecolor] value is 32, for
13307                                  <1>       ; same video mode number.
13308                                  <1>
13309 000107D6 803D[52090000]03        <1>       cmp    byte [vbe3], 3
13310 000107DD 75F2                    <1>       jne    short sysvideo_25 ; Only applicable
13311                                  <1>                                ; for VBE3 video hardware!
13312 000107DF B320                    <1>       mov    bl, 32
13313                                  <1> sysvideo_22_8:
13314 000107E1 881D[B7850100]          <1>       mov    [truecolor], bl     ; 32bpp (00RRGGBBh)
13315                                  <1>       ;mov   [u.r0], bl
13316                                  <1>       ;jmp   short sysvideo_25
13317 000107E7 EBE2                    <1>       jmp    short sysvideo_22_10
13318                                  <1>
```

```
13319                                  <1> sysvideo_22_9:
13320                                  <1>       ; BL = 9
13321                                  <1>       ; Return default true color bpp
13322 000107E9 8A1D[B7850100]           <1>       mov    bl, [truecolor]
13323 000107EF EBDA                     <1>       jmp    short sysvideo_22_10
13324                                  <1> ;sysvideo_22_10:
13325                                  <1>       ;mov   [u.r0], bl
13326                                  <1>       ;jmp   sysret
13327                                  <1>
13328                                  <1> sysvideo_26:
13329                                  <1>       ; 23/12/2020
13330                                  <1>       ; BH = 10
13331                                  <1>       ; Map video memory to user's buffer
13332                                  <1>       ; (multiuser/owner r/w permisions are ignored
13333                                  <1>       ;  for current TRDOS 386 version !)
13334                                  <1>
13335 000107F1 6681E100F0               <1>       and    cx, ~4095  ; clear low 12 bits
13336 000107F6 09C9                     <1>       or     ecx, ecx ; start address of user's buffer
13337 000107F8 74D7                     <1>       jz     short sysvideo_25 ; error !
13338                                  <1>
13339 000107FA 80FB01                   <1>       cmp    bl, 1  ; VGA memory mapping ?
13340 000107FD 740E                     <1>       je     short sysvideo_26_1
13341 000107FF 7718                     <1>       ja     short sysvideo_26_2
13342                                  <1> sysvideo_26_0:
13343                                  <1>       ; BL = 0 : CGA memory (0B8000h) map (32K)
13344 00010801 B800800B00               <1>       mov    eax, 0B8000h
13345 00010806 BB00800000               <1>       mov    ebx, 32768
13346 0001080B EB37                     <1>       jmp    short sysvideo_26_3
13347                                  <1> sysvideo_26_1:
13348                                  <1>       ; BL = 1 : VGA memory (0A0000h) map (64K)
13349 0001080D B800000A00               <1>       mov    eax, 0A0000h
13350 00010812 BB00000100               <1>       mov    ebx, 65536
13351 00010817 EB2B                     <1>       jmp    short sysvideo_26_3
13352                                  <1> sysvideo_26_2:
13353                                  <1>       ; BL = 2 : SVGA memory (LFB) map to user's buffer
13354 00010819 803D[52090000]02         <1>       cmp    byte [vbe3], 2  ; VESA VBE 2/3 vbios ready ?
13355 00010820 72AF                     <1>       jb     short sysvideo_25  ; no, error !
13356 00010822 6681E200F0               <1>       and    dx, ~4095 ; clear low 12 bits
13357 00010827 09D2                     <1>       or     edx, edx  ; buffer size in bytes
13358 00010829 74A6                     <1>       jz     short sysvideo_25  ; error
13359 0001082B 89D3                     <1>       mov    ebx, edx
13360 0001082D A1[2C120300]             <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
13361 00010832 21C0                     <1>       and    eax, eax
13362 00010834 7425                     <1>       jz     short sysvideo_26_5
13363                                  <1>                           ; (LFB parms are not set yet)
13364 00010836 3B1D[30120300]           <1>       cmp    ebx, [LFB_SIZE] ; [LFB_Info+LFBINFO.LFB_size]
13365 0001083C 7606                     <1>       jna    short sysvideo_26_3
13366 0001083E 8B1D[30120300]           <1>       mov    ebx, [LFB_SIZE]
13367                                  <1> sysvideo_26_3:
13368 00010844 52                       <1>       push   edx
13369 00010845 53                       <1>       push   ebx    ; buffer size in bytes
13370 00010846 51                       <1>       push   ecx    ; user's buffer address
13371 00010847 87D9                     <1>       xchg   ebx, ecx
13372 00010849 C1E90C                   <1>       shr    ecx, 12 ; convert buffer size to page count
13373 0001084C E87D5DFFFF               <1>       call   direct_memory_access
13374 00010851 59                       <1>       pop    ecx    ; user's buffer address
13375 00010852 5B                       <1>       pop    ebx    ; buffer size
13376 00010853 5A                       <1>       pop    edx
13377                                  <1>       ;jc    short sysvideo_25 ; error !
13378                                  <1>                            ; [u.r0] = 0
13379                                  <1>       ; 28/02/2021
13380 00010854 7234                     <1>       jc     short sysvideo_27_0 ; error !
13381                                  <1>
13382                                  <1> ;sysvideo_26_4:
13383                                  <1>       ;mov   ebp, [u.usp] ; ebp points to user's registers
13384                                  <1>       ;mov   [ebp+20], edx ; return to user with EDX value
13385                                  <1>       ;mov   [ebp+16], ebx ; EBX
13386                                  <1>       ;mov   [ebp+24], ecx ; ECX
13387                                  <1>       ; eax = physical address of video memory (LFB)
13388                                  <1>       ;mov   [u.r0], eax
13389                                  <1>       ;jmp   sysret
13390 00010856 E919FCFFFF               <1>       jmp    sysvideo_26_4
13391                                  <1>
13392                                  <1> sysvideo_26_5:
13393 0001085B 66A1[E10E0000]           <1>       mov    ax, [def_LFB_addr] ; default LFB for mode 118h
13394                                  <1>       ; ah must be 0C0h or 0D0h or E0h
13395                                  <1>       ; others are nonsence !?
13396 00010861 08E4                     <1>       or     ah, ah
13397                                  <1>       ;jz    short sysvideo_25 ; invalid lfb addr or
13398                                  <1>                           ; it is not a vbe2 -bochs emu-
13399                                  <1>                           ; or vbe3 -real- video bios
13400                                  <1>       ; 28/02/2021
13401 00010863 7425                     <1>       jz     short sysvideo_27_0 ; invalid LFB address
13402                                  <1>
13403 00010865 80FCF0                   <1>       cmp    ah, 0F0h
13404                                  <1>       ;jnb   short sysvideo_25 ; nonsence !?
13405                                  <1>       ; 28/02/2021
13406 00010868 7320                     <1>       jnb    short sysvideo_27_0 ; nonsence !?
13407                                  <1>
13408 0001086A C1E010                   <1>       shl    eax, 16
13409                                  <1>       ;jz    short sysvideo_25 ; eax = 0
13410                                  <1>
13411 0001086D 81FB00907E00             <1>       cmp    ebx, 1920*1080*4 ; maximum value of possible
13412                                  <1>                             ; buffer sizes
13413 00010873 76CF                     <1>       jna    short sysvideo_26_3  ; buffer size is proper
13414                                  <1>       ; resize buffer to fit 4GB limit
13415 00010875 BB00907E00               <1>       mov    ebx, 1920*1080*4
13416 0001087A EBC8                     <1>       jmp    short sysvideo_26_3
13417                                  <1>
13418                                  <1> sysvideo_27:
13419                                  <1>       ; 16/02/2021
13420                                  <1>       ; 18/01/2021
13421 0001087C 80FF0C                   <1>       cmp    bh, 12
13422 0001087F 0F8778010000             <1>       ja     sysvideo_28 ; 19/01/2021
13423                                  <1>
```

```
13424                                  <1>       ; BH = 12
13425                                  <1>       ; Font sub functions.
13426                                  <1>       ; 12/02/2021
13427                                  <1>       ; 11/01/2021
13428                                  <1>       ; 10/01/2021
13429                                  <1>       ;   BL = 0 : Disable system font overwrite
13430                                  <1>       ;   BL = 1 : Enable system font overwrite
13431                                  <1>       ;   BL = 2 : Read system font 8x8
13432                                  <1>       ;   BL = 3 : Read system font 8x14
13433                                  <1>       ;   BL = 4 : Read system font 8x16
13434                                  <1>       ;   BL = 5 : Read user defined font 8x8
13435                                  <1>       ;   BL = 6 : Read user defined font 8x16
13436                                  <1>       ;   BL = 7 : Write system font 8x8
13437                                  <1>       ;   BL = 8 : Write system font 8x14
13438                                  <1>       ;   BL = 9 : Write system font 8x16
13439                                  <1>       ;   BL = 10 : Write user defined font 8x8
13440                                  <1>       ;   BL = 11 : Write user defined font 8x16
13441                                  <1>       ;
13442                                  <1>       ;  BL > 11 : invalid (not implemented)
13443                                  <1>       ;
13444                                  <1>       ; For BL = 1 to 11
13445                                  <1>       ;   ECX = number of characters (<= 256)
13446                                  <1>       ;   EDX = first character (ascii code in DL)
13447                                  <1>       ;   ESI = user's buffer address
13448                                  <1>       ;
13449                                  <1>       ; Return: EAX = character count
13450                                  <1>
13451 00010885 80FB0B                  <1>       cmp    bl, 11
13452 00010888 7605                    <1>       jna    short sysvideo_27_1
13453                                  <1> sysvideo_27_0:
13454 0001088A E93DCFFFFF              <1>       jmp    sysret ; not implemented yet !
13455                                  <1> sysvideo_27_1:
13456 0001088F 66B80001                <1>       mov    ax, 256
13457 00010893 08DB                    <1>       or     bl, bl
13458 00010895 750E                    <1>       jnz    short sysvideo_27_3
13459                                  <1>
13460                                  <1>       ; bl = 0
13461                                  <1>       ; disable system font overwrite
13462                                  <1>
13463 00010897 8025[7E120300]7F        <1>       and    byte [ufont], 7Fh ; clear bit 7
13464                                  <1> sysvideo_27_2:
13465                                  <1>       ;mov   word [u.r0], 256  ; > 0 -> successful
13466 0001089E A3[64030300]            <1>       mov    [u.r0], eax ; 256
13467 000108A3 EBE5                    <1>       jmp    short sysvideo_27_0
13468                                  <1> sysvideo_27_3:
13469 000108A5 80FB01                  <1>       cmp    bl, 1
13470 000108A8 7710                    <1>       ja     short sysvideo_27_4
13471                                  <1>
13472                                  <1>       ; bl = 1
13473                                  <1>       ; enable system font overwrite
13474                                  <1>       ;      if [multi_tasking]= 0 and [u.uid] = 0
13475                                  <1>
13476                                  <1>       ;cmp   byte [multi_tasking], 0
13477                                  <1>       ;                    ; multi tasking enabled ?
13478                                  <1>       ;ja    short sysvideo_27_0 ; yes
13479                                  <1>       ;; 19/01/2021
13480                                  <1>       ;; system maintenance or single user mode
13481                                  <1>       ;cmp   byte [u.uid], 0  ; root ?
13482                                  <1>       ;ja    short sysvideo_27_0 ; no
13483                                  <1>
13484                                  <1>       ; 19/01/2021
13485                                  <1>       ; multi tasking & root check
13486 000108AA E8ECFEFFFF              <1>       call   sysvideo_22_4
13487 000108AF 73D9                    <1>       jnc    short sysvideo_27_0 ; not permitted
13488                                  <1>
13489                                  <1>       ; [multi_tasking]= 0 and [u.uid] = 0
13490                                  <1>
13491 000108B1 800D[7E120300]80        <1>       or     byte [ufont], 80h ; set bit 7
13492                                  <1>
13493 000108B8 EBE4                    <1>       jmp    short sysvideo_27_2
13494                                  <1>
13495                                  <1> sysvideo_27_4:
13496 000108BA 09C9                    <1>       or     ecx, ecx
13497 000108BC 74CC                    <1>       jz     short sysvideo_27_0
13498 000108BE 21D2                    <1>       and    edx, edx
13499 000108C0 7410                    <1>       jz     short sysvideo_27_4_0
13500                                  <1>       ;mov   ax, 256
13501 000108C2 39C1                    <1>       cmp    ecx, eax ; 256
13502 000108C4 77C4                    <1>       ja     short sysvideo_27_0
13503 000108C6 48                      <1>       dec    eax
13504 000108C7 39C2                    <1>       cmp    edx, eax ; 255
13505 000108C9 77BF                    <1>       ja     short sysvideo_27_0
13506 000108CB 40                      <1>       inc    eax
13507 000108CC 29D0                    <1>       sub    eax, edx ; 256 - DX
13508 000108CE 39C8                    <1>       cmp    eax, ecx
13509 000108D0 72B8                    <1>       jb     short sysvideo_27_0
13510                                  <1>
13511                                  <1> sysvideo_27_4_0:
13512 000108D2 89F5                    <1>       mov    ebp, esi
13513                                  <1>
13514 000108D4 80FB06                  <1>       cmp    bl, 6
13515 000108D7 776C                    <1>       ja     short sysvideo_27_13
13516 000108D9 7210                    <1>       jb     short sysvideo_27_5
13517                                  <1>       ; bl = 6
13518 000108DB F605[7E120300]10        <1>       test   byte [ufont], 16 ; 8x16 user font loaded ?
13519 000108E2 74A6                    <1>       jz     short sysvideo_27_0
13520                                  <1>       ; read 8x16 user defined font
13521 000108E4 BE00400900              <1>       mov    esi, VGAFONT16USER
13522 000108E9 EB0C                    <1>       jmp    short sysvideo_27_6
13523                                  <1> sysvideo_27_5:
13524 000108EB 80FB04                  <1>       cmp    bl, 4
13525 000108EE 723D                    <1>       jb     short sysvideo_27_11
13526 000108F0 7723                    <1>       ja     short sysvideo_27_9
13527                                  <1>       ; bl = 4
13528                                  <1>       ; read 8x16 system font
```

```
13529 000108F2 BE[84750100]        <1>       mov   esi, vgafont16
13530                              <1> sysvideo_27_6:
13531                              <1>       ; read 8x16 font
13532 000108F7 66C1E204            <1>       shl   dx, 4 ; * 16
13533 000108FB 66C1E104            <1>       shl   cx, 4 ; * 16 ; 16 bytes per char
13534                              <1> sysvideo_27_7:
13535 000108FF 89EF                <1>       mov   edi, ebp
13536                              <1>       ;add  edi, edx ; 16/02/2021
13537 00010901 01D6                <1>       add   esi, edx
13538                              <1>       ; ecx = byte count
13539                              <1>       ; esi = source (in system memory)
13540                              <1>       ; edi = destination (in user memory)
13541 00010903 E81C110000          <1>       call  transfer_to_user_buffer
13542 00010908 7206                <1>       jc    short sysvideo_27_8
13543 0001090A 890D[64030300]      <1>       mov   [u.r0], ecx
13544                              <1> sysvideo_27_8:
13545 00010910 E9B7CEFFFF          <1>       jmp   sysret
13546                              <1> sysvideo_27_9:
13547                              <1>       ; bl = 5
13548 00010915 F605[7E120300]08    <1>       test  byte [ufont], 8 ; 8x8 user font loaded ?
13549 0001091C 74F2                <1>       jz    short sysvideo_27_8
13550                              <1>       ; read 8x8 user defined font
13551 0001091E BE00500900          <1>       mov   esi, VGAFONT8USER
13552                              <1> sysvideo_27_10:
13553                              <1>       ; read 8x8 font
13554 00010923 66C1E203            <1>       shl   dx, 3 ; * 8
13555 00010927 66C1E103            <1>       shl   cx, 3 ; * 8 ; 8 bytes per char
13556 0001092B EBD2                <1>       jmp   short sysvideo_27_7
13557                              <1>
13558                              <1> sysvideo_27_11:
13559 0001092D 80FB03              <1>       cmp   bl, 3  ; 8x14 system font
13560 00010930 720C                <1>       jb    short sysvideo_27_12 ; 8x8 system font
13561                              <1>       ; bl = 3
13562                              <1>       ; read 8x14 system font
13563                              <1>       ;mov  al, 14
13564                              <1>       ;mul  dl
13565                              <1>       ;mov  dx, ax
13566                              <1>       ;push edx
13567                              <1>       ;mov  ax, 14
13568                              <1>       ;mul  cx
13569                              <1>       ;mov  cx, ax
13570                              <1>       ;pop  edx
13571 00010932 E8A9000000          <1>       call  sysvideo_27_14
13572 00010937 BE[84670100]        <1>       mov   esi, vgafont14
13573 0001093C EBC1                <1>       jmp   short sysvideo_27_7
13574                              <1>
13575                              <1> sysvideo_27_12:
13576                              <1>       ; bl = 2
13577                              <1>       ; read 8x8 system font
13578 0001093E BE[845F0100]        <1>       mov   esi, vgafont8
13579 00010943 EBDE                <1>       jmp   short sysvideo_27_10
13580                              <1>
13581                              <1> sysvideo_27_13:
13582                              <1>       ; overwrite font
13583 00010945 80FB0A              <1>       cmp   bl, 10
13584 00010948 7772                <1>       ja    short sysvideo_27_22 ; 8x16 user font
13585 0001094A 7224                <1>       jb    short sysvideo_27_15
13586                              <1>       ; bl = 10
13587 0001094C BF00500900          <1>       mov   edi, VGAFONT8USER
13588 00010951 F605[7E120300]08    <1>       test  byte [ufont], 8 ; 8x8 user font loaded ?
13589 00010958 7558                <1>       jnz   short sysvideo_27_21 ; yes
13590 0001095A 08ED                <1>       or    ch, ch ; cx = 256
13591                              <1>       ;jnz  short sysvideo_27_21 ; 256 chars
13592 0001095C 7406                <1>       jz    short sysvideo_27_13_0
13593 0001095E 66B90008            <1>       mov   cx, 8*256
13594 00010962 EB37                <1>       jmp   short sysvideo_27_18_0
13595                              <1> sysvideo_27_13_0:
13596                              <1>       ; copy system font to user font before overwrite
13597 00010964 BE[845F0100]        <1>       mov   esi, vgafont8
13598                              <1>       ;push edi
13599                              <1>       ;push ecx
13600                              <1>       ;mov  cl, 64
13601                              <1>       ;rep  movsd
13602                              <1>       ;pop  ecx
13603                              <1>       ;pop  edi
13604                              <1>       ;mov  esi, ebp ; user's font buffer
13605 00010969 E884000000          <1>       call  sysvideo_27_23
13606 0001096E EB42                <1>       jmp   short sysvideo_27_21
13607                              <1>
13608                              <1> sysvideo_27_15:
13609                              <1>       ; check system font overwrite permission
13610 00010970 F605[7E120300]80    <1>       test  byte [ufont], 80h
13611 00010977 7497                <1>       jz    short sysvideo_27_8
13612                              <1>
13613 00010979 80FB08              <1>       cmp   bl, 8
13614 0001097C 773E                <1>       ja    short sysvideo_27_22 ; 8x16 system font
13615 0001097E 722D                <1>       jb    short sysvideo_27_20 ; 8x8 system font
13616                              <1>       ; bl = 8
13617                              <1>       ; overwrite 8x14 system font
13618                              <1>       ;mov  al, 14
13619                              <1>       ;mul  dl
13620                              <1>       ;mov  dx, ax
13621                              <1>       ;push edx
13622                              <1>       ;mov  ax, 14
13623                              <1>       ;mul  cx
13624                              <1>       ;mov  cx, ax
13625                              <1>       ;pop  edx
13626 00010980 E85B000000          <1>       call  sysvideo_27_14
13627 00010985 BF[84670100]        <1>       mov   edi, vgafont14
13628 0001098A EB0D                <1>       jmp   short sysvideo_27_18
13629                              <1> sysvideo_27_16:
13630                              <1>       ; bl = 9
13631                              <1>       ; overwrite 8x16 system font
13632 0001098C BF[84750100]        <1>       mov   edi, vgafont16
13633                              <1> sysvideo_27_17:
```

```
13634                                  <1>       ; overwrite 8x16 font
13635 00010991 66C1E204                <1>       shl    dx, 4 ; * 16
13636 00010995 66C1E104                <1>       shl    cx, 4 ; * 16 ; 16 bytes per char
13637                                  <1> sysvideo_27_18:
13638 00010999 01D7                    <1>       add    edi, edx
13639                                  <1>       ;add   esi, edx ; 16/02/2021
13640                                  <1> sysvideo_27_18_0:
13641                                  <1>       ; ecx = byte count
13642                                  <1>       ; esi = source (in user memory)
13643                                  <1>       ; edi = destination (in system memory)
13644 0001099B E8CE100000              <1>       call   transfer_from_user_buffer
13645 000109A0 7206                    <1>       jc     short sysvideo_27_19
13646 000109A2 890D[64030300]          <1>       mov    [u.r0], ecx
13647                                  <1> sysvideo_27_19:
13648 000109A8 E91FCEFFFF              <1>       jmp    sysret
13649                                  <1> sysvideo_27_20:
13650                                  <1>       ; bl = 7
13651                                  <1>       ; overwrite 8x8 system font
13652 000109AD BF[845F0100]            <1>       mov    edi, vgafont8
13653                                  <1> sysvideo_27_21:
13654                                  <1>       ; write 8x8 font
13655 000109B2 66C1E203                <1>       shl    dx, 3 ; * 8
13656 000109B6 66C1E103                <1>       shl    cx, 3 ; * 8 ; 8 bytes per char
13657 000109BA EBDD                    <1>       jmp    short sysvideo_27_18
13658                                  <1> sysvideo_27_22:
13659                                  <1>       ; bl = 11
13660                                  <1>       ; overwrite 8x16 user defined font
13661 000109BC BF00400900              <1>       mov    edi, VGAFONT16USER
13662 000109C1 F605[7E120300]10        <1>       test   byte [ufont], 16 ; 8x16 user font loaded ?
13663 000109C8 75C7                    <1>       jnz    short sysvideo_27_17 ; yes
13664 000109CA 08ED                    <1>       or     ch, ch ; cx = 256
13665                                  <1>       ;jnz   short sysvideo_27_17 ; 256 chars
13666 000109CC 7406                    <1>       jz     short sysvideo_27_22_0
13667 000109CE 66B90010                <1>       mov    cx, 16*256
13668 000109D2 EBC7                    <1>       jmp    short sysvideo_27_18_0
13669                                  <1> sysvideo_27_22_0:
13670                                  <1>       ; copy system font to user font before overwrite
13671 000109D4 BE[84750100]            <1>       mov    esi, vgafont16
13672                                  <1>       ;push  edi
13673                                  <1>       ;push  ecx
13674                                  <1>       ;mov   cl, 64
13675                                  <1>       ;rep   movsd
13676                                  <1>       ;pop   ecx
13677                                  <1>       ;pop   edi
13678                                  <1>       ;mov   esi, ebp ; user's font buffer
13679 000109D9 E814000000              <1>       call   sysvideo_27_23
13680 000109DE EBB1                    <1>       jmp    short sysvideo_27_17
13681                                  <1>
13682                                  <1> sysvideo_27_14:
13683                                  <1>       ; 16/02/2021
13684 000109E0 52                      <1>       push   edx
13685 000109E1 66B80E00                <1>       mov    ax, 14
13686 000109E5 66F7E1                  <1>       mul    cx
13687 000109E8 89C1                    <1>       mov    ecx, eax
13688 000109EA 5A                      <1>       pop    edx
13689 000109EB B00E                    <1>       mov    al, 14
13690 000109ED F6E2                    <1>       mul    dl
13691 000109EF 89C2                    <1>       mov    edx, eax
13692 000109F1 C3                      <1>       retn
13693                                  <1>
13694                                  <1>       ;mov   al, 14
13695                                  <1>       ;mul   dl
13696                                  <1>       ;mov   dx, ax
13697                                  <1>       ;push  edx
13698                                  <1>       ;; 12/02/2021
13699                                  <1>       ;mov   ax, 14
13700                                  <1>       ;;mov  eax, 14
13701                                  <1>       ;;mul  cx
13702                                  <1>       ;mul   ecx
13703                                  <1>       ;;mov  cx, ax
13704                                  <1>       ;mov   ecx, eax
13705                                  <1>       ;pop   edx
13706                                  <1>       ;retn
13707                                  <1>
13708                                  <1> sysvideo_27_23:
13709 000109F2 57                      <1>       push   edi
13710 000109F3 51                      <1>       push   ecx
13711 000109F4 B140                    <1>       mov    cl, 64
13712 000109F6 F3A5                    <1>       rep    movsd
13713 000109F8 59                      <1>       pop    ecx
13714 000109F9 5F                      <1>       pop    edi
13715 000109FA 89EE                    <1>       mov    esi, ebp ; user's font buffer
13716 000109FC C3                      <1>       retn
13717                                  <1>
13718                                  <1> sysvideo_28:
13719                                  <1>       ; 24/01/2021
13720                                  <1>       ; 23/01/2021
13721                                  <1>       ; 18/01/2021
13722 000109FD 80FF0E                  <1>       cmp    bh, 14
13723 00010A00 0F8275010000            <1>       jb     sysvideo_29
13724 00010A06 0F8754020000            <1>       ja     sysvideo_30
13725                                  <1>
13726                                  <1>       ; BH = 14
13727                                  <1>       ; Save/Restore Super VGA video state
13728                                  <1>
13729                                  <1>       ; BL = options
13730                                  <1>       ;      bit 0 - Save (0) or Restore (1)
13731                                  <1>       ;      bit 1 - controller hardware state
13732                                  <1>       ;      bit 2 - BIOS data state
13733                                  <1>       ;      bit 3 - DAC state
13734                                  <1>       ;      bit 4 - (extended) Register state
13735                                  <1>       ;      bit 5 - system (0) or user (1) memory
13736                                  <1>       ;      bit 6 - verify without transfer
13737                                  <1>       ;      bit 7 - not used (must be 0)
13738                                  <1>
```

```
13739                                  <1>       ; ECX = Buffer address or VideoStateID
13740                                  <1>
13741 00010A0C 803D[52090000]02        <1>       cmp    byte [vbe3], 2 ;  VESA VBE2 or VBE3 ?
13742 00010A13 7717                    <1>       ja     short sysvideo_28_0 ; yes
13743 00010A15 7210                    <1>       jb     short sysvideo_28_16 ; not a SVGA sys !
13744                                  <1>
13745                                  <1>       ; == VBE2 ==
13746                                  <1>       ; Check Bochs/Qemu/VirtualBox PC emulator
13747                                  <1>       ; (vbe2 is usable only for emulator's vbios)
13748 00010A17 8A25[53090000]          <1>       mov    ah, [vbe2bios]
13749 00010A1D 80FCC0                  <1>       cmp    ah, 0C0h
13750 00010A20 7205                    <1>       jb     short sysvideo_28_16 ; unknown vbios !
13751 00010A22 80FCC5                  <1>       cmp    ah, 0C5h
13752 00010A25 7605                    <1>       jna    short sysvideo_28_0
13753                                  <1>              ; Use kernel's vbios functions (video.s)
13754                                  <1> sysvideo_28_16:
13755                                  <1>       ; unknown vbios !
13756 00010A27 E9A0CDFFFF              <1>       jmp    sysret
13757                                  <1>
13758                                  <1> sysvideo_28_0:
13759 00010A2C 80FB7F                  <1>       cmp    bl, 7Fh
13760 00010A2F 77F6                    <1>       ja     short sysvideo_28_16 ; unknown options
13761                                  <1>
13762 00010A31 88DA                    <1>       mov    dl, bl
13763 00010A33 80E21F                  <1>       and    dl, 1Fh
13764 00010A36 D0EA                    <1>       shr    dl, 1
13765 00010A38 74ED                    <1>       jz     short sysvideo_28_16 ; invalid !
13766                                  <1>       ; DL = VBE Function 4F04h Save/Restore options
13767                                  <1>       ;  bit 0 : controller hardware state
13768                                  <1>       ;  bit 1 : BIOS data state
13769                                  <1>       ;  bit 2 : DAC state
13770                                  <1>       ;  bit 3 : (extended) Register state
13771                                  <1>
13772 00010A3A F6C320                  <1>       test   bl, 32 ; bit 5
13773 00010A3D 0F85B1000000            <1>       jnz    sysvideo_28_7 ; user buffer
13774                                  <1>
13775                                  <1>       ; source or destination is kernel/system buffer
13776                                  <1>
13777 00010A43 803D[80120300]00        <1>       cmp    byte [srvsf], 0 ; srs permission flag
13778 00010A4A 76DB                    <1>       jna    short sysvideo_28_16 ; not permitted
13779                                  <1>
13780 00010A4C F6C301                  <1>       test   bl, 1
13781 00010A4F 743A                    <1>       jz     short sysvideo_28_4 ; Save
13782                                  <1>
13783                                  <1>       ; Restore
13784 00010A51 3B0D[82120300]          <1>       cmp    ecx, [VideoStateID]
13785 00010A57 75CE                    <1>       jne    short sysvideo_28_16 ; not correct ID !
13786                                  <1>
13787 00010A59 0FB6CA                  <1>       movzx  ecx, dl
13788 00010A5C 80CA80                  <1>       or     dl, 80h
13789 00010A5F 3A15[81120300]          <1>       cmp    dl, [srvso]
13790 00010A65 75C0                    <1>       jne    short sysvideo_28_16 ; not correct !
13791                                  <1>
13792 00010A67 88DA                    <1>       mov    dl, bl
13793                                  <1>
13794                                  <1>       ; ecx = cl = options
13795 00010A69 E86431FFFF              <1>       call   vbe_srs_gbs
13796                                  <1>       ; ebx = state buffer size (data size)
13797                                  <1>
13798 00010A6E 891D[64030300]          <1>       mov    [u.r0], ebx
13799                                  <1>
13800 00010A74 F6C240                  <1>       test   dl, 64 ; verify without transfer
13801 00010A77 75AE                    <1>       jnz    short sysvideo_28_16 ; yes
13802                                  <1>
13803 00010A79 BE00580900              <1>       mov    esi, VBE3VIDEOSTATE
13804 00010A7E BF00760900              <1>       mov    edi, VBE3SAVERESTOREBLOCK
13805 00010A83 87CB                    <1>       xchg   ecx, ebx
13806 00010A85 F3A4                    <1>       rep    movsb
13807                                  <1>
13808 00010A87 88D9                    <1>       mov    cl, bl
13809                                  <1>
13810                                  <1>       ; 23/01/2021
13811 00010A89 EB44                    <1>       jmp    short sysvideo_28_10
13812                                  <1>
13813                                  <1> sysvideo_28_4:
13814 00010A8B 53                      <1>       push   ebx
13815                                  <1>       ; 24/01/2021
13816 00010A8C 31DB                    <1>       xor    ebx, ebx ; 0 ; use kernel's buffer
13817 00010A8E 881D[81120300]          <1>       mov    [srvso], bl ; 0 ; invalidate
13818 00010A94 891D[82120300]          <1>       mov    [VideoStateID], ebx ; 0 ; invalidate
13819 00010A9A 0FB6CA                  <1>       movzx  ecx, dl ; options
13820 00010A9D B201                    <1>       mov    dl, 1 ; save state
13821 00010A9F E890000000              <1>       call   sysvideo_28_11 ; 23/01/2021
13822                                  <1>       ; Note: VBE3 BIOS data save option will be
13823                                  <1>       ;      disabled.. ; 24/01/2021
13824 00010AA4 89CA                    <1>       mov    edx, ecx ; state (save) options
13825 00010AA6 5B                      <1>       pop    ebx
13826                                  <1>
13827 00010AA7 6683F84F                <1>       cmp    ax, 4Fh ; successful ?
13828 00010AAB 7536                    <1>       jne    short sysvideo_28_3 ; no !
13829                                  <1>
13830 00010AAD F6C340                  <1>       test   bl, 64 ; verify without transfer
13831 00010AB0 7536                    <1>       jnz    short sysvideo_28_6 ; yes
13832                                  <1>
13833                                  <1>       ; ecx = cl = options
13834 00010AB2 E81B31FFFF              <1>       call   vbe_srs_gbs
13835                                  <1>       ; ebx = state buffer size (data size)
13836                                  <1>
13837 00010AB7 BE00760900              <1>       mov    esi, VBE3SAVERESTOREBLOCK
13838 00010ABC BF00580900              <1>       mov    edi, VBE3VIDEOSTATE
13839 00010AC1 89D9                    <1>       mov    ecx, ebx
13840 00010AC3 F3A4                    <1>       rep    movsb
13841                                  <1>
13842 00010AC5 88D1                    <1>       mov    cl, dl
13843 00010AC7 80C980                  <1>       or     cl, 80h ; SVGA (VESA VBE) flag
```

```
13844                                  <1>       ;mov   [srvso], dl
13845                                  <1>
13846 00010ACA E908010000              <1>       jmp    sysvideo_28_15
13847                                  <1>
13848                                  <1>       ; 23/01/2021
13849                                  <1> sysvideo_28_10:
13850                                  <1>       ; CL = VESA VBE3 Save/Restore options
13851                                  <1>
13852 00010ACF B202                    <1>       mov    dl, 2 ; restore state
13853                                  <1>
13854 00010AD1 E85C000000              <1>       call   sysvideo_28_1
13855                                  <1>
13856 00010AD6 6683F84F                <1>       cmp    ax, 4Fh ; successful ?
13857 00010ADA 7407                    <1>       je     short sysvideo_28_3
13858                                  <1>       ;jmp   short sysvideo_28_9
13859                                  <1>
13860                                  <1> sysvideo_28_9:
13861                                  <1>       ; return zero size (error) to user
13862 00010ADC 29C0                    <1>       sub    eax, eax
13863                                  <1> sysvideo_28_5:
13864 00010ADE A3[64030300]            <1>       mov    [u.r0], eax
13865                                  <1> sysvideo_28_3:
13866 00010AE3 E9E4CCFFFF              <1>       jmp    sysret
13867                                  <1>
13868                                  <1> sysvideo_28_6:
13869                                  <1>       ; use timer ticks as VideoStateID
13870 00010AE8 A1[70890100]            <1>       mov    eax, [TIMER_LH]
13871 00010AED 09C0                    <1>       or     eax, eax
13872 00010AEF 75ED                    <1>       jnz    short sysvideo_28_5
13873 00010AF1 40                      <1>       inc    eax
13874 00010AF2 EBEA                    <1>       jmp    short sysvideo_28_5
13875                                  <1>
13876                                  <1> sysvideo_28_7:
13877                                  <1>       ; save/restore to/from user buffer
13878                                  <1>
13879                                  <1>       ; 23/01/2021
13880 00010AF4 89CE                    <1>       mov    esi, ecx ; user's vstate buffer
13881 00010AF6 BF00760900              <1>       mov    edi, VBE3SAVERESTOREBLOCK
13882                                  <1>
13883 00010AFB 0FB6CA                  <1>       movzx  ecx, dl ; VESA VBE func 4F04h options
13884                                  <1>
13885                                  <1>       ; source or destination is user buffer
13886 00010AFE F6C301                  <1>       test   bl, 1
13887 00010B01 7444                    <1>       jz     short sysvideo_28_12  ; Save
13888                                  <1>
13889                                  <1>       ; Restore
13890 00010B03 803D[80120300]00        <1>       cmp    byte [srvsf], 0 ; srs permission flag
13891 00010B0A 766A                    <1>       jna    short sysvideo_28_14 ; not permitted
13892                                  <1>
13893 00010B0C 88DA                    <1>       mov    dl, bl ; 'sysvideo' options
13894                                  <1>
13895                                  <1>       ; ecx = cl = options
13896 00010B0E E8BF30FFFF              <1>       call   vbe_srs_gbs
13897                                  <1>       ; ebx = state buffer size (data size)
13898                                  <1>
13899 00010B13 891D[64030300]          <1>       mov    [u.r0], ebx ; transfer count
13900                                  <1>
13901 00010B19 F6C240                  <1>       test   dl, 64 ; verify without transfer
13902 00010B1C 7558                    <1>       jnz    short sysvideo_28_14 ; yes
13903                                  <1>
13904 00010B1E 6681FB0008              <1>       cmp    bx, 2048
13905 00010B23 73B7                    <1>       jnb    short sysvideo_28_9 ; invalid
13906                                  <1>
13907 00010B25 87CB                    <1>       xchg   ecx, ebx
13908                                  <1>       ; esi = user buffer
13909                                  <1>       ; edi = VBE3SAVERESTOREBLOCK
13910                                  <1>
13911 00010B27 E8420F0000              <1>       call   transfer_from_user_buffer
13912 00010B2C 72AE                    <1>       jc     short sysvideo_28_9 ; error
13913                                  <1>
13914 00010B2E 89D9                    <1>       mov    ecx, ebx ; Function 4F04h options
13915 00010B30 EB9D                    <1>       jmp    short sysvideo_28_10 ; 23/01/2021
13916                                  <1>
13917                                  <1> sysvideo_28_1:
13918 00010B32 31DB                    <1>       xor    ebx, ebx ; 0 ; use kernel's buffer
13919                                  <1> sysvideo_28_11:
13920                                  <1>       ; 24/01/2021
13921 00010B34 803D[52090000]03        <1>       cmp    byte [vbe3], 3
13922 00010B3B 7405                    <1>       je     short sysvideo_28_2
13923                                  <1>
13924                                  <1>       ; VESA VBE2 (BOCHS/QEMU/VBOX) video bios
13925 00010B3D E9F92FFFFF              <1>       jmp    _vbe_biosfn_save_restore_state
13926                                  <1> sysvideo_28_2:
13927                                  <1>       ;24/01/2021
13928                                  <1>       ;mov   eax, 4F04h  ; Save/Restore vstate
13929                                  <1>       ; VESA VBE3 video bios
13930 00010B42 E9890EFFFF              <1>       jmp    _vbe3_pmfn_save_restore_state
13931                                  <1>
13932                                  <1> sysvideo_28_12:
13933                                  <1>       ; Save
13934                                  <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
13935                                  <1>
13936                                  <1>       ;movzx ecx, dl ; options
13937 00010B47 56                      <1>       push   esi
13938 00010B48 53                      <1>       push   ebx
13939                                  <1>       ; 23/01/2021
13940 00010B49 B201                    <1>       mov    dl, 1 ; save state
13941 00010B4B E8E2FFFFFF              <1>       call   sysvideo_28_1
13942 00010B50 5A                      <1>       pop    edx ; 'sysvideo' options
13943 00010B51 5F                      <1>       pop    edi ; user's video state buffer
13944                                  <1>
13945 00010B52 6683F84F                <1>       cmp    ax, 4Fh ; successful ?
13946 00010B56 751E                    <1>       jne    short sysvideo_28_14 ; no !
13947                                  <1>
13948                                  <1>       ; ecx = cl = options
```

```
13949 00010B58 E87530FFFF          <1>       call   vbe_srs_gbs
13950                              <1>       ; ebx = state buffer size (data size)
13951                              <1>
13952 00010B5D 89D9                <1>       mov    ecx, ebx ; transfer count
13953                              <1>
13954 00010B5F F6C240              <1>       test   dl, 64 ; verify without transfer
13955 00010B62 750C                <1>       jnz    short sysvideo_28_13 ; yes
13956                              <1>
13957                              <1>       ;mov   edi, esi
13958 00010B64 BE00760900          <1>       mov    esi, VBE3SAVERESTOREBLOCK
13959 00010B69 E8B60E0000          <1>       call   transfer_to_user_buffer
13960 00010B6E 7206                <1>       jc     short sysvideo_28_14
13961                              <1> sysvideo_28_13:
13962 00010B70 890D[64030300]      <1>       mov    [u.r0], ecx
13963                              <1> sysvideo_28_14:
13964 00010B76 E951CCFFFF          <1>       jmp    sysret
13965                              <1>
13966                              <1> sysvideo_29:
13967                              <1>       ; 18/01/2021
13968                              <1>       ; BH = 13
13969                              <1>       ; Save/Restore std VGA video state
13970                              <1>
13971                              <1>       ; bl = 0..3
13972                              <1>       ;      save to or restore from
13973                              <1>       ;      system buffer, VBE3VIDEOSTATE
13974                              <1>       ;      ECX = VideoStateID for restoring
13975                              <1>       ; bl = 4..7
13976                              <1>       ;      save to or restore from
13977                              <1>       ;      user buffer pointed by ECX
13978                              <1>
13979 00010B7B 80FB03              <1>       cmp    bl, 3
13980 00010B7E 7776                <1>       ja     short sysvideo_29_6
13981                              <1>
13982                              <1>       ; source or destination is kernel/system buffer
13983                              <1>
13984 00010B80 803D[80120300]00    <1>       cmp    byte [srvsf], 0 ; srs permission flag
13985 00010B87 7668                <1>       jna    short sysvideo_29_5 ; not permitted
13986                              <1>
13987 00010B89 F6C301              <1>       test   bl, 1
13988 00010B8C 7437                <1>       jz     short sysvideo_29_2  ; Save
13989                              <1>
13990                              <1>       ; Restore
13991 00010B8E 3B0D[82120300]      <1>       cmp    ecx, [VideoStateID]
13992 00010B94 755B                <1>       jne    short sysvideo_29_5 ; not correct ID !
13993 00010B96 80FB01              <1>       cmp    bl, 1
13994 00010B99 7709                <1>       ja     short sysvideo_29_0
13995                              <1>       ; bl = 1
13996 00010B9B BB6E000000          <1>       mov    ebx, 110
13997 00010BA0 B103                <1>       mov    cl, 3 ; ctrl, vbios data
13998 00010BA2 EB07                <1>       jmp    short sysvideo_29_1
13999                              <1> sysvideo_29_0:
14000                              <1>       ; bl  = 3
14001 00010BA4 BB72030000          <1>       mov    ebx, 882
14002 00010BA9 B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
14003                              <1> sysvideo_29_1:
14004 00010BAB 3A0D[81120300]      <1>       cmp    cl, [srvso]
14005 00010BB1 753E                <1>       jne    short sysvideo_29_5 ; not correct !
14006                              <1>
14007 00010BB3 BE00580900          <1>       mov    esi, VBE3VIDEOSTATE ; 22/01/2021
14008 00010BB8 E8AF31FFFF          <1>       call   biosfn_restore_video_state
14009 00010BBD 891D[64030300]      <1>       mov    [u.r0], ebx ; video state size (bytes)
14010                              <1>       ;jmp   sysret
14011 00010BC3 EB2C                <1>       jmp    short sysvideo_29_5
14012                              <1> sysvideo_29_2:
14013                              <1>       ;mov   esi, ecx
14014 00010BC5 BF00580900          <1>       mov    edi, VBE3VIDEOSTATE
14015                              <1>
14016 00010BCA B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
14017 00010BCC 08DB                <1>       or     bl, bl
14018 00010BCE 7502                <1>       jnz    short sysvideo_29_3 ; bl = 2
14019                              <1>       ; bl = 0
14020 00010BD0 B103                <1>       mov    cl, 3 ; ctrl, vbios data
14021                              <1> sysvideo_29_3:
14022 00010BD2 E82730FFFF          <1>       call   biosfn_save_video_state
14023                              <1> sysvideo_28_15:
14024                              <1>       ; use timer ticks as VideoStateID
14025 00010BD7 A1[70890100]        <1>       mov    eax, [TIMER_LH]
14026 00010BDC 21C0                <1>       and    eax, eax
14027 00010BDE 7501                <1>       jnz    short sysvideo_29_4
14028 00010BE0 40                  <1>       inc    eax
14029                              <1> sysvideo_29_4:
14030 00010BE1 880D[81120300]      <1>       mov    [srvso], cl
14031 00010BE7 A3[82120300]        <1>       mov    [VideoStateID], eax
14032 00010BEC A3[64030300]        <1>       mov    [u.r0], eax
14033                              <1> sysvideo_29_5:
14034 00010BF1 E9D6CBFFFF          <1>       jmp    sysret
14035                              <1>
14036                              <1> sysvideo_29_6:
14037 00010BF6 80FB07              <1>       cmp    bl, 7
14038 00010BF9 77F6                <1>       ja     short sysvideo_29_5 ; invalid sub function
14039                              <1>
14040 00010BFB 89CE                <1>       mov    esi, ecx
14041 00010BFD BF00760900          <1>       mov    edi, VBE3SAVERESTOREBLOCK
14042                              <1>
14043                              <1>       ; source or destination is user buffer
14044 00010C02 F6C301              <1>       test   bl, 1
14045 00010C05 7434                <1>       jz     short sysvideo_29_9  ; Save
14046                              <1>
14047                              <1>       ; Restore
14048 00010C07 803D[80120300]00    <1>       cmp    byte [srvsf], 0 ; srs permission flag
14049 00010C0E 76E1                <1>       jna    short sysvideo_29_5 ; not permitted
14050                              <1>
14051                              <1>       ;mov   esi, ecx
14052                              <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
14053                              <1>
```

```
14054 00010C10 80FB07              <1>       cmp    bl, 7
14055 00010C13 7409                <1>       je     short sysvideo_29_7
14056                              <1>       ; bl = 5
14057 00010C15 B303                <1>       mov    bl, 3
14058 00010C17 B96E000000          <1>       mov    ecx, 110
14059 00010C1C EB05                <1>       jmp    short sysvideo_29_8
14060                              <1> sysvideo_29_7:
14061                              <1>       ; bl = 7
14062 00010C1E B972030000          <1>       mov    ecx, 882
14063                              <1> sysvideo_29_8:
14064 00010C23 E8460E0000          <1>       call   transfer_from_user_buffer
14065 00010C28 72C7                <1>       jc     short sysvideo_29_5
14066 00010C2A 890D[64030300]      <1>       mov    [u.r0], ecx
14067 00010C30 88D9                <1>       mov    cl, bl ; mov cl,7 (mov cl,3)
14068 00010C32 89FE                <1>       mov    esi, edi ; VBE3SAVERESTOREBLOCK
14069                              <1>       ; cl = 3 or 7
14070 00010C34 E83331FFFF          <1>       call   biosfn_restore_video_state
14071 00010C39 EBB6                <1>       jmp    sysvideo_29_5
14072                              <1>       ;jmp   sysret
14073                              <1> sysvideo_29_9:
14074                              <1>       ; Save
14075                              <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
14076                              <1>
14077 00010C3B 80FB06              <1>       cmp    bl, 6
14078 00010C3E 7409                <1>       je     short sysvideo_29_10
14079                              <1>       ; bl = 4
14080 00010C40 BB6E000000          <1>       mov    ebx, 110
14081 00010C45 B103                <1>       mov    cl, 3 ; ctrl, vbios data
14082 00010C47 EB07                <1>       jmp    short sysvideo_29_11
14083                              <1> sysvideo_29_10:
14084                              <1>       ; bl = 6
14085 00010C49 BB72030000          <1>       mov    ebx, 882
14086 00010C4E B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
14087                              <1> sysvideo_29_11:
14088 00010C50 E8A92FFFFF          <1>       call   biosfn_save_video_state
14089                              <1>
14090 00010C55 89D9                <1>       mov    ecx, ebx ; transfer count
14091 00010C57 89F7                <1>       mov    edi, esi
14092 00010C59 BE00760900          <1>       mov    esi, VBE3SAVERESTOREBLOCK
14093                              <1>
14094                              <1>       ;call  transfer_to_user_buffer
14095                              <1>       ;jc    short sysvideo_29_5
14096                              <1>       ;mov   [u.r0], ecx ; transfer count
14097                              <1>       ;;jmp  sysret
14098                              <1>       ;jmp   short sysvideo_29_5
14099                              <1>
14100 00010C5E EB1A                <1>       jmp    short sysvideo_29_12
14101                              <1>
14102                              <1> sysvideo_30:
14103 00010C60 80FF0F              <1>       cmp    bh, 15
14104 00010C63 7722                <1>       ja     short sysvideo_31 ; invalid function
14105                              <1>
14106                              <1>       ; BH = 15
14107                              <1>       ; Copy VESA EDID to user's buffer
14108                              <1>
14109 00010C65 803D[92420000]4F    <1>       cmp    byte [edid], 4Fh
14110 00010C6C 7519                <1>       jne    short sysvideo_31 ; not ready !
14111                              <1>
14112                              <1>       ;and   ecx, ecx
14113                              <1>       ;jz    short sysvideo_31
14114                              <1>
14115                              <1>       ; ecx = user's buffer address
14116 00010C6E 89CF                <1>       mov    edi, ecx
14117 00010C70 BE[9C110300]        <1>       mov    esi, edid_info
14118 00010C75 B980000000          <1>       mov    ecx, 128 ; 128 bytes
14119                              <1> sysvideo_29_12:
14120 00010C7A E8A50D0000          <1>       call   transfer_to_user_buffer
14121 00010C7F 7206                <1>       jc     short sysvideo_31
14122                              <1>
14123 00010C81 890D[64030300]      <1>       mov    [u.r0], ecx ; EDID size, 128 bytes
14124                              <1> sysvideo_31:
14125 00010C87 E940CBFFFF          <1>       jmp    sysret
14126                              <1>
14127                              <1> mkdir:
14128                              <1>       ; 04/12/2015 (14 byte directory names)
14129                              <1>       ; 12/10/2015
14130                              <1>       ; 17/06/2015 (Retro UNIX 386 v1 - Beginning)
14131                              <1>       ; 29/04/2013 - 01/08/2013 (Retro UNIX 8086 v1)
14132                              <1>       ;
14133                              <1>       ; 'mkdir' makes a directory entry from the name pointed to
14134                              <1>       ; by u.namep into the current directory.
14135                              <1>       ;
14136                              <1>       ; INPUTS ->
14137                              <1>       ;    u.namep - points to a file name
14138                              <1>       ;                that is about to be a directory entry.
14139                              <1>       ;    ii - current directory's i-number.
14140                              <1>       ; OUTPUTS ->
14141                              <1>       ;    u.dirbuf+2 - u.dirbuf+10 - contains file name.
14142                              <1>       ;    u.off - points to entry to be filled
14143                              <1>       ;           in the current directory
14144                              <1>       ;    u.base - points to start of u.dirbuf.
14145                              <1>       ;    r1 - contains i-number of current directory
14146                              <1>       ;
14147                              <1>       ; ((AX = R1)) output
14148                              <1>       ;
14149                              <1>       ;    (Retro UNIX Prototype : 11/11/2012, UNIXCOPY.ASM)
14150                              <1>         ;    ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
14151                              <1>       ;
14152                              <1>
14153                              <1>       ; 17/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
14154 00010C8C 31C0                <1>       xor    eax, eax
14155 00010C8E BF[9A030300]        <1>       mov     edi, u.dirbuf+2
14156 00010C93 89FE                <1>       mov    esi, edi
14157 00010C95 AB                  <1>       stosd
14158 00010C96 AB                  <1>       stosd
```

```
14159                                  <1>       ; 04/12/2015 (14 byte directory names)
14160 00010C97 AB                      <1>       stosd
14161 00010C98 66AB                    <1>       stosw
14162                                  <1>             ; jsr r0,copyz; u.dirbuf+2; u.dirbuf+10. / clear this
14163 00010C9A 89F7                    <1>       mov   edi, esi ; offset to u.dirbuf
14164                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
14165                                  <1>       ;mov   ebp, [u.namep]
14166 00010C9C E80D030000              <1>       call  trans_addr_nmbp ; convert virtual address to physical
14167                                  <1>             ; esi = physical address (page start + offset)
14168                                  <1>             ; ecx = byte count in the page (1 - 4096)
14169                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
14170                                  <1>             ; mov u.namep,r2 / r2 points to name of directory entry
14171                                  <1>             ; mov $u.dirbuf+2,r3 / r3 points to u.dirbuf+2
14172                                  <1> mkdir_1: ; 1:
14173 00010CA1 45                      <1>       inc   ebp ; 12/10/2015
14174                                  <1>       ;
14175                                  <1>       ; / put characters in the directory name in u.dirbuf+2 - u.dirbuf+10
14176                                  <1>        ; 01/08/2013
14177 00010CA2 AC                      <1>       lodsb
14178                                  <1>             ; movb (r2)+,r1 / move character in name to r1
14179 00010CA3 20C0                    <1>       and   al, al
14180 00010CA5 7427                    <1>       jz    short mkdir_3
14181                                  <1>             ; beq 1f / if null, done
14182 00010CA7 3C2F                    <1>       cmp   al, '/'
14183                                  <1>             ; cmp r1,$'/ / is it a "/"?
14184 00010CA9 7414                    <1>       je    short mkdir_err
14185                                  <1>       ;je   error
14186                                  <1>             ; beq error9 / yes, error
14187                                  <1>       ; 12/10/2015
14188 00010CAB 6649                    <1>       dec   cx
14189 00010CAD 7505                    <1>       jnz   short mkdir_2
14190                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
14191 00010CAF E800030000              <1>       call  trans_addr_nm ; convert virtual address to physical
14192                                  <1>             ; esi = physical address (page start + offset)
14193                                  <1>             ; ecx = byte count in the page
14194                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
14195                                  <1> mkdir_2:
14196 00010CB4 81FF[A8030300]          <1>       cmp    edi, u.dirbuf+16 ; ; 04/12/2015 (10 -> 16)
14197                                  <1>             ; cmp r3,$u.dirbuf+10. / have we reached the last slot for
14198                                  <1>                                    ; / a char?
14199 00010CBA 74E5                    <1>       je    short mkdir_1
14200                                  <1>             ; beq 1b / yes, go back
14201 00010CBC AA                      <1>       stosb
14202                                  <1>             ; movb r1,(r3)+ / no, put the char in the u.dirbuf
14203 00010CBD EBE2                    <1>       jmp   short mkdir_1
14204                                  <1>             ; br 1b / get next char
14205                                  <1> mkdir_err:
14206                                  <1>       ; 17/06/2015
14207 00010CBF C705[C8030300]1300-     <1>       mov   dword [u.error], ERR_NOT_DIR ; 'not a valid directory !'
14207 00010CC7 0000                    <1>
14208 00010CC9 E9DECAFFFF              <1>       jmp   error
14209                                  <1>
14210                                  <1> mkdir_3: ; 1:
14211 00010CCE A1[78030300]            <1>       mov   eax, [u.dirp]
14212 00010CD3 A3[80030300]            <1>       mov   [u.off], eax
14213                                  <1>             ; mov u.dirp,u.off / pointer to empty current directory
14214                                  <1>                                ; / slot to u.off
14215                                  <1> wdir: ; 29/04/2013
14216 00010CD8 C705[84030300]-         <1>         mov     dword [u.base], u.dirbuf
14216 00010CDE [98030300]              <1>
14217                                  <1>             ; mov $u.dirbuf,u.base / u.base points to created file name
14218 00010CE2 C705[88030300]1000-     <1>         mov     dword [u.count], 16 ; 04/12/2015 (10 -> 16)
14218 00010CEA 0000                    <1>
14219                                  <1>             ; mov $10.,u.count / u.count = 10
14220 00010CEC 66A1[51040300]          <1>       mov   ax, [ii]
14221                                  <1>             ; mov ii,r1 / r1 has i-number of current directory
14222 00010CF2 B201                    <1>       mov   dl, 1 ; owner flag mask ; RETRO UNIX 8086 v1 modification !
14223 00010CF4 E8741D0000              <1>       call  access
14224                                  <1>             ; jsr r0,access; 1 / get i-node and set its file up
14225                                  <1>                                ; / for writing
14226                                  <1>       ; AX = i-number of current directory
14227                                  <1>       ; 01/08/2013
14228 00010CF9 FE05[C6030300]          <1>       inc    byte [u.kcall] ; the caller is 'mkdir' sign
14229 00010CFF E8820F0000              <1>       call  writei
14230                                  <1>             ; jsr r0,writei / write into directory
14231 00010D04 C3                      <1>       retn
14232                                  <1>             ; rts r0
14233                                  <1>
14234                                  <1> sysexec:
14235                                  <1>       ; 18/11/2017
14236                                  <1>       ; 14/11/2017
14237                                  <1>       ; 13/11/2017
14238                                  <1>       ; 24/10/2016, 04/01/2017
14239                                  <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
14240                                  <1>       ; 23/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
14241                                  <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
14242                                  <1>       ;
14243                                  <1>       ; 'sysexec' initiates execution of a file whose path name if
14244                                  <1>       ; pointed to by 'name' in the sysexec call.
14245                                  <1>       ; 'sysexec' performs the following operations:
14246                                  <1>       ;    1. obtains i-number of file to be executed via 'namei'.
14247                                  <1>       ;    2. obtains i-node of file to be exceuted via 'iget'.
14248                                  <1>       ;    3. sets trap vectors to system routines.
14249                                  <1>       ;    4. loads arguments to be passed to executing file into
14250                                  <1>       ;      highest locations of user's core
14251                                  <1>       ;    5. puts pointers to arguments in locations immediately
14252                                  <1>       ;      following arguments.
14253                                  <1>       ;    6.      saves number of arguments in next location.
14254                                  <1>       ;    7. intializes user's stack area so that all registers
14255                                  <1>       ;      will be zeroed and the PS is cleared and the PC set
14256                                  <1>       ;      to core when 'sysret' restores registers
14257                                  <1>       ;      and does an rti.
14258                                  <1>       ;    8. inializes u.r0 and u.sp
14259                                  <1>       ;    9. zeros user's core down to u.r0
14260                                  <1>       ;   10.      reads executable file from storage device into core
```

```
14261                                 <1>       ;      starting at location 'core'.
14262                                 <1>       ;   11.      sets u.break to point to end of user's code with
14263                                 <1>       ;      data area appended.
14264                                 <1>       ;   12.      calls 'sysret' which returns control at location
14265                                 <1>       ;      'core' via 'rti' instruction.
14266                                 <1>       ;
14267                                 <1>       ; Calling sequence:
14268                                 <1>       ;      sysexec; namep; argp
14269                                 <1>       ; Arguments:
14270                                 <1>       ;      namep - points to pathname of file to be executed
14271                                 <1>       ;      argp  - address of table of argument pointers
14272                                 <1>       ;      argp1... argpn - table of argument pointers
14273                                 <1>       ;      argp1:<...0> ... argpn:<...0> - argument strings
14274                                 <1>       ; Inputs: (arguments)
14275                                 <1>       ; Outputs: -
14276                                 <1>       ; ..........................................................
14277                                 <1>       ;
14278                                 <1>       ; Retro UNIX 386 v1 modification:
14279                                 <1>       ;      User application runs in it's own virtual space
14280                                 <1>       ;      which is izolated from kernel memory (and other
14281                                 <1>       ;      memory pages) via 80386   paging in ring 3
14282                                 <1>       ;      privilige mode. Virtual start address is always 0.
14283                                 <1>       ;      User's core memory starts at linear address 400000h
14284                                 <1>       ;      (the end of the 1st 4MB).
14285                                 <1>       ;
14286                                 <1>       ; Retro UNIX 8086 v1 modification:
14287                                 <1>       ;      user/application segment and system/kernel segment
14288                                 <1>       ;      are different and sysenter/sysret/sysrele routines
14289                                 <1>       ;      are different (user's registers are saved to
14290                                 <1>       ;      and then restored from system's stack.)
14291                                 <1>       ;
14292                                 <1>       ;      NOTE: Retro UNIX 8086 v1 'arg2' routine gets these
14293                                 <1>       ;            arguments which were in these registers;
14294                                 <1>       ;            but, it returns by putting the 1st argument
14295                                 <1>       ;            in 'u.namep' and the 2nd argument
14296                                 <1>       ;            on top of stack. (1st argument is offset of the
14297                                 <1>       ;            file/path name in the user's program segment.)
14298                                 <1>
14299                                 <1>       ;call  arg2
14300                                 <1>       ; * name - 'u.namep' points to address of file/path name
14301                                 <1>       ;          in the user's program segment ('u.segmnt')
14302                                 <1>       ;          with offset in BX register (as sysopen argument 1).
14303                                 <1>       ; * argp - sysexec argument 2 is in CX register
14304                                 <1>       ;          which is on top of stack.
14305                                 <1>       ;
14306                                 <1>             ; jsr r0,arg2 / arg0 in u.namep,arg1 on top of stack
14307                                 <1>
14308                                 <1>       ; 23/06/2015 (32 bit modifications)
14309                                 <1>
14310                                 <1>       ;; 13/11/2017
14311                                 <1>       ;;mov  [u.namep], ebx ; argument 1
14312                                 <1>         ; 18/10/2015
14313 00010D05 890D[4C040300]        <1>       mov     [argv], ecx  ; * ; argument 2
14314                                 <1>
14315                                 <1>       ; 13/11/2017
14316 00010D0B 89DE                  <1>       mov    esi, ebx
14317 00010D0D E88E210000            <1>       call   set_working_path_x
14318 00010D12 7319                  <1>       jnc    short sysexec_0
14319                                 <1>
14320                                 <1>       ;; 'bad command or file name'
14321                                 <1>       ;mov   eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
14322                                 <1>
14323                                 <1>       ; 'file not found !' error
14324 00010D14 B802000000            <1>       mov    eax, ERR_NOT_FOUND ; 02h ; TRDOS 8086
14325                                 <1> sysexec_not_found_err:
14326                                 <1> sysexec_access_error:
14327                                 <1> sysexec_ext_error:
14328 00010D19 A3[64030300]          <1>       mov    [u.r0], eax
14329 00010D1E A3[C8030300]          <1>       mov    [u.error], eax
14330 00010D23 E84D220000            <1>       call   reset_working_path
14331 00010D28 E97FCAFFFF            <1>       jmp    error
14332                                 <1>
14333                                 <1> sysexec_0:
14334                                 <1>       ; 13/11/2017
14335                                 <1>       ;mov   esi, FindFile_Name
14336 00010D2D 66B80018              <1>         mov  ax, 1800h ; Only files
14337 00010D31 E81B86FFFF            <1>       call   find_first_file
14338 00010D36 72E1                  <1>       jc     short sysexec_not_found_err ; eax = 2
14339                                 <1>
14340                                 <1>       ; check_ file attributes
14341                                 <1>       ; (attribute bits = 00ADVSHR) ; 18h = Directory+Volume
14342                                 <1>       ; BL = Attributes byte
14343                                 <1>
14344 00010D38 F6C306                <1>         test bl, 6  ; system file or hidden file (S+H)
14345                                 <1>       ;jz    short sysexec_0ext
14346 00010D3B 7417                  <1>       jz     short sysexec_1 ; yes
14347                                 <1>
14348                                 <1>       ; 13/11/2017
14349                                 <1>       ; /// TRDOS386 permission check for multiuser mode ///
14350                                 <1>       ; SYSTEM file or HIDDEN file !!
14351                                 <1>       ; (Only super user has permission to run this file.)
14352                                 <1>
14353                                 <1>       ; ([u.uid]=0 for super user or root in multiuser mode)
14354                                 <1>       ; ([u.uid]=0 for any users in singleuser mode)
14355 00010D3D 803D[B0030300]00      <1>       cmp    byte [u.uid], 0 ; Super User ([u.uid]=0) ?
14356                                 <1>       ;jna   short sysexec_0ext
14357 00010D44 760E                  <1>       jna    short sysexec_1 ; yes
14358                                 <1>
14359                                 <1>       ; 'permission denied !' error
14360 00010D46 B80B000000            <1>         mov  eax, ERR_FILE_ACCESS  ; 11 = ERR_PERM_DENIED
14361 00010D4B EBCC                  <1>         jmp  short sysexec_access_error
14362                                 <1>
14363                                 <1> sysexec_not_exf:
14364                                 <1>       ; 'not executable file !' error
14365 00010D4D B816000000            <1>       mov    eax, ERR_NOT_EXECUTABLE
```

```
14366 00010D52 EBC5               <1>       jmp    sysexec_ext_error
14367                             <1>
14368                             <1> ;sysexec_0ext:
14369                             <1> sysexec_1:
14370                             <1>       ; 18/11/2017
14371 00010D54 BE[98920100]       <1>       mov    esi, FindFile_Name
14372                             <1>       ; 13/11/2017
14373                             <1>       ; check program file name extension
14374                             <1>       ; ('.PRG' for current TRDOS version)
14375 00010D59 E896A0FFFF         <1>       call   check_prg_filename_ext
14376 00010D5E 72ED               <1>       jc     short sysexec_not_exf
14377                             <1>
14378                             <1>       ; 18/11/2017
14379 00010D60 3C50               <1>       cmp    al, 'P'
14380 00010D62 75E9               <1>       jne    short sysexec_not_exf
14381                             <1>
14382                             <1>       ; '.PRG' extension is OK.
14383                             <1>       ; Only '.PRG' files are valid program files
14384                             <1>       ; for current TRDOS 386 version.
14385                             <1>
14386 00010D64 8B15[C4920100]     <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
14387 00010D6A 66A1[BC920100]     <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
14388 00010D70 C1E010             <1>       shl    eax, 16
14389 00010D73 66A1[C2920100]     <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
14390                             <1>       ; EAX = First Cluster number
14391                             <1>       ; EDX = File Size
14392                             <1>
14393 00010D79 A3[51040300]       <1>       mov    [ii], eax
14394 00010D7E 8915[55040300]     <1>       mov    [i.size], edx
14395                             <1>
14396                             <1> ;sysexec_1:
14397                             <1>       ; 13/11/2017 - TRDOS 386 (TRDOS v2.0)
14398                             <1>       ; 24/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
14399                             <1>         ; Moving arguments to the end of [u.upage]
14400                             <1>       ; (by regarding page borders in user's memory space)
14401                             <1>       ;
14402                             <1>       ; 10/10/2015
14403                             <1>       ; 21/07/2015
14404 00010D84 89E5               <1>       mov    ebp, esp ; (**)
14405                             <1>       ; 18/10/2015
14406 00010D86 89EF               <1>       mov    edi, ebp
14407 00010D88 B900010000         <1>       mov    ecx, MAX_ARG_LEN ; 256
14408                             <1>       ;sub   edi, MAX_ARG_LEN ; 256
14409 00010D8D 29CF               <1>       sub    edi, ecx
14410 00010D8F 89FC               <1>       mov    esp, edi ; *!*
14411 00010D91 31C0               <1>       xor    eax, eax
14412 00010D93 A3[8C030300]       <1>       mov    [u.nread], eax ; 0
14413 00010D98 66A3[4A040300]     <1>       mov    [argc], ax ; 0 ; 13/11/2017
14414 00010D9E 49                 <1>       dec    ecx ; 256 - 1
14415 00010D9F 890D[88030300]     <1>       mov    [u.count], ecx ; MAX_ARG_LEN - 1 ; 255
14416                             <1>       ;mov   dword [u.count], MAX_ARG_LEN - 1 ; 255
14417                             <1> sysexec_2:
14418 00010DA5 8B35[4C040300]     <1>       mov    esi, [argv] ; 18/10/2015
14419 00010DAB E866000000         <1>       call   get_argp
14420 00010DB0 B904000000         <1>       mov    ecx, 4 ; mov ecx, 4
14421                             <1> sysexec_3:
14422 00010DB5 21C0               <1>       and    eax, eax
14423 00010DB7 0F846F050000       <1>         jz     sysexec_6
14424                             <1>       ; 18/10/2015
14425 00010DBD 010D[4C040300]     <1>       add    [argv], ecx ; 4
14426 00010DC3 66FF05[4A040300]   <1>       inc    word [argc]
14427                             <1>       ;
14428 00010DCA A3[84030300]       <1>       mov    [u.base], eax
14429                             <1>       ; 23/10/2015
14430 00010DCF 66C705[C4030300]00- <1>      mov    word [u.pcount], 0
14430 00010DD7 00                 <1>
14431                             <1> sysexec_4:
14432 00010DD8 E8E70B0000         <1>       call   cpass ; get a character from user's core memory
14433 00010DDD 750E               <1>         jnz  short sysexec_5
14434                             <1>              ; (max. 255 chars + null)
14435                             <1>       ; 18/10/2015
14436 00010DDF 28C0               <1>       sub    al, al
14437 00010DE1 AA                 <1>       stosb
14438 00010DE2 FF05[8C030300]     <1>       inc    dword [u.nread]
14439 00010DE8 E93F050000         <1>       jmp    sysexec_6 ; 24/04/2016
14440                             <1> sysexec_5:
14441 00010DED AA                 <1>       stosb
14442 00010DEE 20C0               <1>       and    al, al
14443 00010DF0 75E6               <1>       jnz    short sysexec_4
14444 00010DF2 B904000000         <1>       mov    ecx, 4
14445 00010DF7 390D[48040300]     <1>       cmp    [ncount], ecx ; 4
14446 00010DFD 72A6               <1>       jb     short sysexec_2
14447 00010DFF 8B35[44040300]     <1>       mov    esi, [nbase]
14448 00010E05 010D[44040300]     <1>       add    [nbase], ecx ; 4
14449 00010E0B 66290D[48040300]   <1>       sub    [ncount], cx
14450 00010E12 8B06               <1>       mov    eax, [esi]
14451 00010E14 EB9F               <1>       jmp    short sysexec_3
14452                             <1>
14453                             <1> get_argp:
14454                             <1>       ; 14/11/2017 - TRDOS 386 (TRDOS v2.0)
14455                             <1>       ; 18/10/2015 (nbase, ncount)
14456                             <1>       ; 21/07/2015
14457                             <1>       ; 24/06/2015 (Retro UNIX 386 v1)
14458                             <1>       ; Get (virtual) address of argument from user's core memory
14459                             <1>       ;
14460                             <1>       ; INPUT:
14461                             <1>       ;      esi = virtual address of argument pointer
14462                             <1>       ; OUTPUT:
14463                             <1>       ;      eax = virtual address of argument
14464                             <1>       ;
14465                             <1>       ; Modified registers: EAX, EBX, ECX, EDX, ESI
14466                             <1>       ;
14467 00010E16 833D[BC030300]00   <1>       cmp     dword [u.ppgdir], 0 ; /etc/init ?
14468                             <1>                                   ; (the caller is kernel)
14469 00010E1D 7667               <1>         jna     short get_argpk
```

```
14470                                  <1>       ;
14471 00010E1F 89F3                    <1>       mov    ebx, esi
14472 00010E21 E89653FFFF              <1>       call   get_physical_addr ; get physical address
14473 00010E26 0F8289000000            <1>         jc     get_argp_err
14474 00010E2C A3[44040300]            <1>       mov    [nbase], eax ; physical address
14475 00010E31 66890D[48040300]        <1>       mov    [ncount], cx ; remain byte count in page (1-4096)
14476 00010E38 B804000000              <1>       mov    eax, 4 ; 21/07/2015
14477 00010E3D 6639C1                  <1>       cmp    cx, ax ; 4
14478 00010E40 735D                    <1>       jnb    short get_argp2
14479 00010E42 89F3                    <1>       mov    ebx, esi
14480 00010E44 01CB                    <1>       add    ebx, ecx
14481 00010E46 E87153FFFF              <1>       call   get_physical_addr ; get physical address
14482 00010E4B 7268                    <1>       jc     short get_argp_err
14483                                  <1>       ;push  esi
14484 00010E4D 89C6                    <1>       mov    esi, eax
14485 00010E4F 66870D[48040300]        <1>       xchg   cx, [ncount]
14486 00010E56 8735[44040300]          <1>       xchg   esi, [nbase]
14487 00010E5C B504                    <1>       mov    ch, 4
14488 00010E5E 28CD                    <1>       sub    ch, cl
14489                                  <1> get_argp0:
14490 00010E60 AC                      <1>       lodsb
14491 00010E61 6650                    <1>       push   ax
14492 00010E63 FEC9                    <1>       dec    cl
14493 00010E65 75F9                    <1>         jnz    short get_argp0
14494 00010E67 8B35[44040300]          <1>       mov    esi, [nbase]
14495                                  <1>       ; 21/07/2015
14496 00010E6D 0FB6C5                  <1>       movzx  eax, ch
14497 00010E70 0105[44040300]          <1>       add    [nbase], eax
14498 00010E76 662905[48040300]        <1>       sub    [ncount], ax
14499                                  <1> get_argp1:
14500 00010E7D AC                      <1>       lodsb
14501 00010E7E FECD                    <1>       dec    ch
14502 00010E80 7447                    <1>         jz     short get_argp3
14503 00010E82 6650                    <1>         push ax
14504 00010E84 EBF7                    <1>       jmp    short get_argp1
14505                                  <1> get_argpk:
14506                                  <1>       ; Argument is in kernel's memory space
14507 00010E86 66C705[48040300]00-     <1>       mov    word [ncount], PAGE_SIZE ; 4096
14507 00010E8E 10                      <1>
14508 00010E8F 8935[44040300]          <1>       mov    [nbase], esi
14509 00010E95 8305[44040300]04        <1>       add    dword [nbase], 4
14510 00010E9C 8B06                    <1>       mov    eax, [esi] ; virtual addr. = physical addr.
14511 00010E9E C3                      <1>       retn
14512                                  <1> get_argp2:
14513                                  <1>       ; 21/07/2015
14514                                  <1>       ;mov   eax, 4
14515 00010E9F 8B15[44040300]          <1>       mov    edx, [nbase] ; 18/10/2015
14516 00010EA5 0105[44040300]          <1>       add    [nbase], eax
14517 00010EAB 662905[48040300]        <1>       sub    [ncount], ax
14518                                  <1>       ;
14519 00010EB2 8B02                    <1>       mov    eax, [edx]
14520 00010EB4 C3                      <1>       retn
14521                                  <1> get_argp_err:
14522 00010EB5 A3[C8030300]            <1>       mov    [u.error], eax
14523                                  <1>       ; 14/11/2017
14524 00010EBA B801000000              <1>       mov    eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
14525 00010EBF A3[64030300]            <1>       mov    [u.r0], eax
14526 00010EC4 E9E3C8FFFF              <1>       jmp    error
14527                                  <1> get_argp3:
14528 00010EC9 B103                    <1>       mov    cl, 3
14529                                  <1> get_argp4:
14530 00010ECB C1E008                  <1>       shl    eax, 8
14531 00010ECE 665A                    <1>       pop    dx
14532 00010ED0 88D0                    <1>       mov    al, dl
14533 00010ED2 E2F7                    <1>         loop   get_argp4
14534                                  <1>       ;pop   esi
14535 00010ED4 C3                      <1>       retn
14536                                  <1>
14537                                  <1> sysstat:
14538                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
14539                                  <1>       ; temporary !
14540 00010ED5 B801000000              <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
14541 00010EDA A3[C8030300]            <1>         mov    [u.error], eax
14542 00010EDF A3[64030300]            <1>         mov    [u.r0], eax
14543 00010EE4 E9C3C8FFFF              <1>       jmp    error
14544                                  <1>
14545                                  <1> sysfstat:
14546                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
14547                                  <1>       ; temporary !
14548 00010EE9 B801000000              <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
14549 00010EEE A3[C8030300]            <1>         mov    [u.error], eax
14550 00010EF3 A3[64030300]            <1>         mov    [u.r0], eax
14551 00010EF8 E9AFC8FFFF              <1>       jmp    error
14552                                  <1>
14553                                  <1> fclose:
14554                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
14555                                  <1>       ;
14556                                  <1>       ; 18/06/2015 (Retro UNIX 386 v1 - Beginning)
14557                                  <1>       ;            (32 bit offset pointer modification)
14558                                  <1>       ; 19/04/2013 - 12/01/2014 (Retro UNIX 8086 v1)
14559                                  <1>       ;
14560                                  <1>       ; Given the file descriptor (index to the u.fp list)
14561                                  <1>       ; 'fclose' first gets the i-number of the file via 'getf'.
14562                                  <1>       ; If i-node is active (i-number > 0) the entry in
14563                                  <1>       ; u.fp list is cleared. If all the processes that opened
14564                                  <1>       ; that file close it, then fsp etry is freed and the file
14565                                  <1>       ; is closed. If not a return is taken.
14566                                  <1>       ; If the file has been deleted while open, 'anyi' is called
14567                                  <1>       ; to see anyone else has it open, i.e., see if it is appears
14568                                  <1>       ; in another entry in the fsp table. Upon return from 'anyi'
14569                                  <1>       ; a check is made to see if the file is special.
14570                                  <1>       ;
14571                                  <1>       ; INPUTS ->
14572                                  <1>       ;    r1 - contains the file descriptor (value=0,1,2...)
14573                                  <1>       ;    u.fp - list of entries in the fsp table
```

```
14574                                  <1>       ;    fsp - table of entries (4 words/entry) of open files.
14575                                  <1>       ; OUTPUTS ->
14576                                  <1>       ;    r1 - contains the same file descriptor
14577                                  <1>       ;    r2 - contains i-number
14578                                  <1>       ;
14579                                  <1>       ; ((AX = R1))
14580                                  <1>       ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
14581                                  <1>       ;
14582                                  <1>       ; Retro UNIX 8086 v1 modification : CF = 1
14583                                  <1>       ;              if i-number of the file is 0. (error)
14584                                  <1>       ;
14585                                  <1>       ; TRDOS 386 (06/10/2016)
14586                                  <1>       ;
14587                                  <1>       ; INPUT:
14588                                  <1>       ;     EAX = File Handle (File Descriptor, File Index)
14589                                  <1>       ;
14590                                  <1>       ; OUTPUT:
14591                                  <1>       ;     CF = 1 -> File not open !
14592                                  <1>       ;     CF = 0 -> OK!
14593                                  <1>       ;          EBX = File Number (System)
14594                                  <1>       ;          [cdev] = Logical DOS Drive Number
14595                                  <1>       ;          EAX = File Handle/Number (user)
14596                                  <1>       ;
14597                                  <1>       ; Modified Registers: EBX
14598                                  <1>
14599 00010EFD 50                      <1>       push   eax ; File handle
14600                                  <1>
14601 00010EFE E846000000              <1>       call   getf
14602 00010F03 0F8245240000            <1>       jc     device_close ; eax = device number
14603                                  <1>
14604 00010F09 80BB[16990100]01        <1>       cmp    byte [ebx+OF_MODE], 1 ; open mode ; 0 = empty entry
14605 00010F10 722E                    <1>       jb     short fclose_1            ; 1 = read, 2 = write
14606                                  <1>
14607 00010F12 83F801                  <1>       cmp    eax, 1 ; is the first cluster number > 0
14608 00010F15 7229                    <1>       jb     short fclose_1 ; no, this is empty entry
14609                                  <1>
14610                                  <1> fclose_0:
14611 00010F17 FE8B[2A990100]          <1>       dec    byte [ebx+OF_OPENCOUNT] ; decrement the number of processes
14612                                  <1>                                    ; that have opened the file
14613 00010F1D 7921                    <1>       jns    short fclose_1 ; jump if not negative (jump if bit 7 is 0)
14614                                  <1>                     ; if all processes haven't closed the file, return
14615                                  <1>       ;
14616                                  <1>       ; eax ; First cluster
14617 00010F1F 31C0                    <1>       xor    eax, eax ; 0
14618 00010F21 8883[16990100]          <1>       mov    [ebx+OF_MODE], al ; 0 = empty entry
14619                                  <1>       ;mov   [ebx+OF_STATUS], al ; 0 = empty entry
14620 00010F27 66C1E302                <1>       shl    bx, 2
14621 00010F2B 8983[E4980100]          <1>       mov    [ebx+OF_FCLUSTER], eax ; 0
14622 00010F31 8983[FC990100]          <1>       mov    [ebx+OF_CCLUSTER], eax ; 0
14623                                  <1>       ;mov   [ebx+OF_CCINDEX], eax ; 0
14624 00010F37 A3[74030300]            <1>       mov    [u.fofp], eax ; 0
14625 00010F3C 66C1EB02                <1>       shr    bx, 2
14626                                  <1> fclose_1: ; 1:
14627 00010F40 58                      <1>       pop    eax ; File handle (File Descriptor, File Index)
14628 00010F41 C680[6A030300]00        <1>       mov    byte [eax+u.fp], 0 ; clear that entry in the u.fp list
14629 00010F48 C3                      <1>       retn
14630                                  <1>
14631                                  <1> getf:
14632                                  <1>       ; 12/10/2016
14633                                  <1>       ; 11/10/2016
14634                                  <1>       ; 08/10/2016
14635                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
14636                                  <1>       ; / get the device number and the i-number of an open file
14637                                  <1>       ; 13/05/2015
14638                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
14639                                  <1>       ; 19/04/2013 - 18/11/2013 (Retro UNIX 8086 v1)
14640                                  <1>       ;
14641 00010F49 89C3                    <1>       mov    ebx, eax
14642                                  <1> getf1:
14643 00010F4B 83FB0A                  <1>       cmp    ebx, 10
14644 00010F4E 730A                    <1>         jnb  short getf2
14645 00010F50 8A9B[6A030300]          <1>       mov    bl, [ebx+u.fp]
14646 00010F56 08DB                    <1>       or     bl, bl
14647 00010F58 7503                    <1>       jnz    short getf3
14648                                  <1> getf2:
14649                                  <1>       ; 'File not open !' error (ax=0)
14650 00010F5A 29C0                    <1>       sub    eax, eax
14651 00010F5C C3                      <1>       retn
14652                                  <1> getf3:
14653 00010F5D F6C380                  <1>       test   bl, 80h
14654 00010F60 7530                    <1>       jnz    short getf5 ; device
14655 00010F62 FECB                    <1>       dec    bl ; 0 based
14656 00010F64 8A83[0C990100]          <1>       mov    al, [ebx+OF_DRIVE]
14657 00010F6A A2[46030300]            <1>       mov    [cdev], al
14658 00010F6F C0E302                  <1>       shl    bl, 2 ; *4 (dword offset)
14659 00010F72 8B83[5C990100]          <1>       mov    eax, [ebx+OF_SIZE]
14660 00010F78 A3[55040300]            <1>       mov    [i.size], eax ; file size
14661 00010F7D 8D83[34990100]          <1>       lea    eax, [ebx+OF_POINTER] ;12/10/2016
14662 00010F83 A3[74030300]            <1>       mov    [u.fofp], eax
14663 00010F88 8B83[E4980100]          <1>       mov    eax, [ebx+OF_FCLUSTER]
14664 00010F8E C0EB02                  <1>       shr    bl, 2 ; /4 (byte offset)
14665                                  <1> getf4:
14666 00010F91 C3                      <1>       retn
14667                                  <1> getf5:
14668                                  <1>       ; get device number
14669 00010F92 80E37F                  <1>       and    bl, 7Fh ; 1 to 7Fh
14670 00010F95 FECB                    <1>       dec    bl ; 0 based (0 to 7Eh)
14671 00010F97 8A83[3E970100]          <1>       mov    al, [ebx+DEV_DRIVER]
14672 00010F9D 8AAB[A8960100]          <1>       mov    ch, [ebx+DEV_ACCESS]
14673 00010FA3 8A8B[5C970100]          <1>       mov    cl, [ebx+DEV_OPENMODE]
14674 00010FA9 80E5FE                  <1>       and    ch, 0FEh ; reset bit 0 ; dev_close
14675 00010FAC F9                      <1>       stc ; cf = 1
14676 00010FAD C3                      <1>       retn
14677                                  <1>
14678                                  <1> trans_addr_nmbp:
```

```
14679                                 <1>       ; 18/10/2015
14680                                 <1>       ; 12/10/2015
14681 00010FAE 8B2D[7C030300]         <1>       mov    ebp, [u.namep]
14682                                 <1> trans_addr_nm:
14683                                 <1>       ; Convert virtual (pathname) address to physical address
14684                                 <1>       ; (Retro UNIX 386 v1 feature only !)
14685                                 <1>       ; 18/10/2015
14686                                 <1>       ; 12/10/2015 (u.pnbase & u.pncount has been removed from code)
14687                                 <1>       ; 02/07/2015
14688                                 <1>       ; 17/06/2015
14689                                 <1>       ; 16/06/2015
14690                                 <1>       ;
14691                                 <1>       ; INPUTS:
14692                                 <1>       ;      ebp = pathname address (virtual) ; [u.namep]
14693                                 <1>       ;      [u.pgdir] = user's page directory
14694                                 <1>       ; OUTPUT:
14695                                 <1>       ;       esi = physical address of the pathname
14696                                 <1>       ;      ecx = remain byte count in the page
14697                                 <1>       ;
14698                                 <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI)
14699                                 <1>       ;
14700 00010FB4 833D[BC030300]00       <1>         cmp     dword [u.ppgdir], 0  ; /etc/init ? (sysexec)
14701 00010FBB 7618                   <1>       jna    short trans_addr_nmk ; the caller is os kernel;
14702                                 <1>                                   ; it is already physical address
14703 00010FBD 50                     <1>       push   eax
14704 00010FBE 89EB                   <1>       mov    ebx, ebp ; [u.namep] ; pathname address (virtual)
14705 00010FC0 E8F751FFFF             <1>              call   get_physical_addr ; get physical address
14706 00010FC5 7204                   <1>       jc     short tr_addr_nm_err
14707                                 <1>       ; 18/10/2015
14708                                 <1>       ; eax = physical address
14709                                 <1>       ; cx = remain byte count in page (1-4096)
14710                                 <1>              ; 12/10/2015 (cx = [u.pncount])
14711 00010FC7 89C6                   <1>       mov    esi, eax ; 12/10/2015 (esi=[u.pnbase])
14712 00010FC9 58                     <1>       pop    eax
14713 00010FCA C3                     <1>       retn
14714                                 <1>
14715                                 <1> tr_addr_nm_err:
14716 00010FCB A3[C8030300]           <1>       mov    [u.error], eax
14717                                 <1>       ;pop   eax
14718 00010FD0 E9D7C7FFFF             <1>       jmp    error
14719                                 <1>
14720                                 <1> trans_addr_nmk:
14721                                 <1>       ; 12/10/2015
14722                                 <1>       ; 02/07/2015
14723 00010FD5 8B35[7C030300]         <1>       mov    esi, [u.namep]  ; [u.pnbase]
14724 00010FDB 66B90010               <1>       mov    cx, PAGE_SIZE ; 4096 ; [u.pncount]
14725 00010FDF C3                     <1>       retn
14726                                 <1>
14727                                 <1> sysbreak:
14728                                 <1>       ; 18/10/2015
14729                                 <1>       ; 07/10/2015
14730                                 <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
14731                                 <1>       ; 20/06/2013 - 24/03/2014 (Retro UNIX 8086 v1)
14732                                 <1>       ;
14733                                 <1>       ; 'sysbreak' sets the programs break points.
14734                                 <1>       ; It checks the current break point (u.break) to see if it is
14735                                 <1>       ; between "core" and the stack (sp). If it is, it is made an
14736                                 <1>       ; even address (if it was odd) and the area between u.break
14737                                 <1>       ; and the stack is cleared. The new breakpoint is then put
14738                                 <1>       ; in u.break and control is passed to 'sysret'.
14739                                 <1>       ;
14740                                 <1>       ; Calling sequence:
14741                                 <1>       ;      sysbreak; addr
14742                                 <1>       ; Arguments: -
14743                                 <1>       ;
14744                                 <1>       ; Inputs: u.break - current breakpoint
14745                                 <1>       ; Outputs: u.break - new breakpoint
14746                                 <1>       ;      area between old u.break and the stack (sp) is cleared.
14747                                 <1>       ; .........................................................
14748                                 <1>       ;
14749                                 <1>       ; Retro UNIX 8086 v1 modification:
14750                                 <1>       ;      The user/application program puts breakpoint address
14751                                 <1>       ;       in BX register as 'sysbreak' system call argument.
14752                                 <1>       ;      (argument transfer method 1)
14753                                 <1>       ;
14754                                 <1>       ;  NOTE: Beginning of core is 0 in Retro UNIX 8086 v1 !
14755                                 <1>       ;      ((!'sysbreak' is not needed in Retro UNIX 8086 v1!))
14756                                 <1>       ;  NOTE:
14757                                 <1>       ;      'sysbreak' clears extended part (beyond of previous
14758                                 <1>       ;      'u.break' address) of user's memory for original unix's
14759                                 <1>       ;      'bss' compatibility with Retro UNIX 8086 v1 (19/11/2013)
14760                                 <1>
14761                                 <1>              ; mov u.break,r1 / move users break point to r1
14762                                 <1>              ; cmp r1,$core / is it the same or lower than core?
14763                                 <1>              ; blos 1f / yes, 1f
14764                                 <1>       ; 23/06/2015
14765 00010FE0 8B2D[90030300]         <1>       mov    ebp, [u.break] ; virtual address (offset)
14766                                 <1>       ;and   ebp, ebp
14767                                 <1>       ;jz    short sysbreak_3
14768                                 <1>       ; Retro UNIX 386 v1 NOTE: u.break points to virtual address !!!
14769                                 <1>       ; (Even break point address is not needed for Retro UNIX 386 v1)
14770 00010FE6 8B15[5C030300]         <1>       mov    edx, [u.sp] ; kernel stack at the beginning of sys call
14771 00010FEC 83C20C                 <1>       add    edx, 12 ; EIP -4-> CS -4-> EFLAGS -4-> ESP (user)
14772                                 <1>       ; 07/10/2015
14773 00010FEF 891D[90030300]         <1>       mov    [u.break], ebx ; virtual address !!!
14774                                 <1>       ;
14775 00010FF5 3B1A                   <1>       cmp    ebx, [edx] ; compare new break point with
14776                                 <1>                         ; with top of user's stack (virtual!)
14777 00010FF7 7323                   <1>       jnb    short sysbreak_3
14778                                 <1>              ; cmp r1,sp / is it the same or higher
14779                                 <1>                      ; / than the stack?
14780                                 <1>              ; bhis 1f / yes, 1f
14781 00010FF9 89DE                   <1>       mov    esi, ebx
14782 00010FFB 29EE                   <1>       sub    esi, ebp ; new break point - old break point
14783 00010FFD 761D                   <1>       jna    short sysbreak_3
```

```
14784                                  <1>       ;push  ebx
14785                                  <1> sysbreak_1:
14786 00010FFF 89EB                    <1>       mov    ebx, ebp
14787 00011001 E8B651FFFF              <1>       call   get_physical_addr ; get physical address
14788 00011006 72C3                    <1>       jc     tr_addr_nm_err
14789                                  <1>       ; 18/10/2015
14790 00011008 89C7                    <1>       mov    edi, eax
14791 0001100A 29C0                    <1>       sub    eax, eax ; 0
14792                                  <1>              ; ECX = remain byte count in page (1-4096)
14793 0001100C 39CE                    <1>       cmp    esi, ecx
14794 0001100E 7302                    <1>       jnb    short sysbreak_2
14795 00011010 89F1                    <1>       mov    ecx, esi
14796                                  <1> sysbreak_2:
14797 00011012 29CE                    <1>       sub    esi, ecx
14798 00011014 01CD                    <1>       add    ebp, ecx
14799 00011016 F3AA                    <1>       rep    stosb
14800 00011018 09F6                    <1>       or     esi, esi
14801 0001101A 75E3                    <1>       jnz    short sysbreak_1
14802                                  <1>       ;
14803                                  <1>              ; bit $1,r1 / is it an odd address
14804                                  <1>              ; beq 2f / no, its even
14805                                  <1>              ; clrb (r1)+ / yes, make it even
14806                                  <1>       ; 2: / clear area between the break point and the stack
14807                                  <1>              ; cmp r1,sp / is it higher or same than the stack
14808                                  <1>              ; bhis 1f / yes, quit
14809                                  <1>              ; clr (r1)+ / clear word
14810                                  <1>              ; br 2b / go back
14811                                  <1>       ;pop   ebx
14812                                  <1> sysbreak_3: ; 1:
14813                                  <1>       ;mov   [u.break], ebx ; virtual address !!!
14814                                  <1>              ; jsr r0,arg; u.break / put the "address"
14815                                  <1>                     ; / in u.break (set new break point)
14816                                  <1>              ; br sysret4 / br sysret
14817 0001101C E9ABC7FFFF              <1>       jmp    sysret
14818                                  <1>
14819                                  <1> sysseek: ; / moves read write pointer in an fsp entry
14820                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
14821                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
14822                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
14823                                  <1>       ;
14824                                  <1>       ; 'sysseek' changes the r/w pointer of (3rd word of in an
14825                                  <1>       ; fsp entry) of an open file whose file descriptor is in u.r0.
14826                                  <1>       ; The file descriptor refers to a file open for reading or
14827                                  <1>       ; writing. The read (or write) pointer is set as follows:
14828                                  <1>       ;     * if 'ptrname' is 0, the pointer is set to offset.
14829                                  <1>       ;     * if 'ptrname' is 1, the pointer is set to its
14830                                  <1>       ;       current location plus offset.
14831                                  <1>       ;     * if 'ptrname' is 2, the pointer is set to the
14832                                  <1>       ;       size of file plus offset.
14833                                  <1>       ; The error bit (e-bit) is set for an undefined descriptor.
14834                                  <1>       ;
14835                                  <1>       ; Calling sequence:
14836                                  <1>       ;     sysseek; offset; ptrname
14837                                  <1>       ; Arguments:
14838                                  <1>       ;     offset - number of bytes desired to move
14839                                  <1>       ;              the r/w pointer
14840                                  <1>       ;     ptrname - a switch indicated above
14841                                  <1>       ;
14842                                  <1>       ; Inputs: r0 - file descriptor
14843                                  <1>       ; Outputs: -
14844                                  <1>       ; ..........................................................
14845                                  <1>       ;
14846                                  <1>       ; Retro UNIX 8086 v1 modification:
14847                                  <1>       ;       'sysseek' system call has three arguments; so,
14848                                  <1>       ;     * 1st argument, file descriptor is in BX (BL) register
14849                                  <1>       ;     * 2nd argument, offset is in CX register
14850                                  <1>       ;     * 3rd argument, ptrname/switch is in DX (DL) register
14851                                  <1>
14852 00011021 E821000000              <1>       call   seektell
14853                                  <1>       ; EAX = Current R/W pointer of the file
14854                                  <1>       ; EBX = [u.fofp]
14855                                  <1>       ; [u.base] = offset (ECX input)
14856                                  <1>
14857 00011026 0305[84030300]          <1>       add    eax, [u.base]
14858 0001102C 8903                    <1>       mov    [ebx], eax
14859 0001102E E999C7FFFF              <1>       jmp    sysret
14860                                  <1>
14861                                  <1> systell: ; / get the r/w pointer
14862                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0) - temporary !-
14863                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
14864                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
14865                                  <1>       ;
14866                                  <1>       ; Retro UNIX 8086 v1 modification:
14867                                  <1>       ; ! 'systell' does not work in original UNIX v1,
14868                                  <1>       ;          it returns with error !
14869                                  <1>       ; Inputs: r0 - file descriptor
14870                                  <1>       ; Outputs: r0 - file r/w pointer
14871                                  <1>
14872                                  <1>       ;xor   ecx, ecx ; 0
14873 00011033 BA01000000              <1>       mov    edx, 1 ; 05/08/2013
14874                                  <1>       ;call  seektell
14875 00011038 E810000000              <1>       call   seektell0 ; 05/08/2013
14876                                  <1>       ;; 06/11/2016
14877                                  <1>       ;; mov eax, [ebx]
14878 0001103D A3[64030300]            <1>       mov    [u.r0], eax
14879 00011042 E985C7FFFF              <1>       jmp    sysret
14880                                  <1>
14881                                  <1> ; Original unix v1 'systell' system call:
14882                                  <1>              ; jsr r0,seektell
14883                                  <1>              ; br error4
14884                                  <1>
14885                                  <1> seektell:
14886                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
14887                                  <1>       ; 03/01/2016
14888                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
```

```
14889                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
14890                                  <1>       ;
14891                                  <1>       ; 'seektell' puts the arguments from sysseek and systell
14892                                  <1>       ; call in u.base and u.count. It then gets the i-number of
14893                                  <1>       ; the file from the file descriptor in u.r0 and by calling
14894                                  <1>       ; getf. The i-node is brought into core and then u.count
14895                                  <1>       ; is checked to see it is a 0, 1, or 2.
14896                                  <1>       ; If it is 0 - u.count stays the same
14897                                  <1>       ;          1 - u.count = offset (u.fofp)
14898                                  <1>       ;        2 - u.count = i.size (size of file)
14899                                  <1>       ;
14900                                  <1>       ; !! Retro UNIX 8086 v1 modification:
14901                                  <1>       ;     Argument 1, file descriptor is in BX;
14902                                  <1>       ;     Argument 2, offset is in CX;
14903                                  <1>       ;     Argument 3, ptrname/switch is in DX register.
14904                                  <1>       ;
14905                                  <1>       ; ((Return -> eax = base for offset (position= base+offset))
14906                                  <1>       ;
14907 00011047 890D[84030300]           <1>       mov    [u.base], ecx ; offset
14908                                  <1> seektell0:
14909 0001104D 8915[88030300]           <1>       mov    [u.count], edx
14910                                  <1>       ; EBX = file descriptor (file number)
14911 00011053 E8F3FEFFFF               <1>       call   getf1
14912                                  <1>       ; EAX = First cluster of the file
14913                                  <1>       ; EBX = File number (Open file number)
14914                                  <1>       ; [u.fofp] = Pointer to File pointer
14915                                  <1>       ; [i.size] = File size
14916                                  <1>
14917 00011058 09C0                    <1>       or     eax, eax
14918 0001105A 7514                    <1>       jnz    short seektell1
14919                                  <1>
14920 0001105C B80A000000              <1>       mov    eax, ERR_FILE_NOT_OPEN
14921 00011061 A3[64030300]            <1>       mov    [u.r0], eax
14922 00011066 A3[C8030300]            <1>       mov    dword [u.error], eax ; 'file not open !'
14923 0001106B E93CC7FFFF              <1>       jmp    error
14924                                  <1>
14925                                  <1> seektell1:
14926 00011070 8B1D[74030300]           <1>         mov     ebx, [u.fofp]
14927 00011076 803D[88030300]01         <1>       cmp    byte [u.count], 1
14928 0001107D 7705                    <1>       ja     short seektell2
14929 0001107F 7409                    <1>       je     short seektell3
14930 00011081 31C0                    <1>       xor    eax, eax
14931 00011083 C3                      <1>       retn
14932                                  <1>
14933                                  <1> seektell2:
14934 00011084 A1[55040300]            <1>         mov        eax, [i.size]
14935 00011089 C3                      <1>       retn
14936                                  <1>
14937                                  <1> seektell3:
14938 0001108A 8B03                    <1>       mov    eax, [ebx]
14939 0001108C C3                      <1>       retn
14940                                  <1>
14941                                  <1> sysintr: ; / set interrupt handling
14942                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
14943                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
14944                                  <1>       ;
14945                                  <1>       ; 'sysintr' sets the interrupt handling value. It puts
14946                                  <1>       ; argument of its call in u.intr then branches into 'sysquit'
14947                                  <1>       ; routine. u.tty is checked if to see if a control tty exists.
14948                                  <1>       ; If one does the interrupt character in the tty buffer is
14949                                  <1>       ; cleared and 'sysret'is called. If one does not exits
14950                                  <1>       ; 'sysret' is just called.
14951                                  <1>       ;
14952                                  <1>       ; Calling sequence:
14953                                  <1>       ;     sysintr; arg
14954                                  <1>       ; Argument:
14955                                  <1>       ;     arg - if 0, interrupts (ASCII DELETE) are ignored.
14956                                  <1>       ;         - if 1, intterupts cause their normal result
14957                                  <1>       ;             i.e force an exit.
14958                                  <1>       ;         - if arg is a location within the program,
14959                                  <1>       ;           control is passed to that location when
14960                                  <1>       ;           an interrupt occurs.
14961                                  <1>       ; Inputs: -
14962                                  <1>       ; Outputs: -
14963                                  <1>       ; ..........................................................
14964                                  <1>       ;
14965                                  <1>       ; Retro UNIX 8086 v1 modification:
14966                                  <1>       ;       'sysintr' system call sets u.intr to value of BX
14967                                  <1>       ;     then branches into sysquit.
14968                                  <1>       ;
14969 0001108D 66891D[AA030300]         <1>       mov    [u.intr], bx
14970                                  <1>             ; jsr r0,arg; u.intr / put the argument in u.intr
14971                                  <1>             ; br 1f / go into quit routine
14972 00011094 E933C7FFFF              <1>       jmp    sysret
14973                                  <1>
14974                                  <1> sysquit:
14975                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
14976                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
14977                                  <1>       ;
14978                                  <1>       ; 'sysquit' turns off the quit signal. it puts the argument of
14979                                  <1>       ; the call in u.quit. u.tty is checked if to see if a control
14980                                  <1>       ; tty exists. If one does the interrupt character in the tty
14981                                  <1>       ; buffer is cleared and 'sysret'is called. If one does not exits
14982                                  <1>       ; 'sysret' is just called.
14983                                  <1>       ;
14984                                  <1>       ; Calling sequence:
14985                                  <1>       ;     sysquit; arg
14986                                  <1>       ; Argument:
14987                                  <1>       ;     arg - if 0, this call diables quit signals from the
14988                                  <1>       ;           typewriter (ASCII FS)
14989                                  <1>       ;         - if 1, quits are re-enabled and cause execution to
14990                                  <1>       ;           cease and a core image to be produced.
14991                                  <1>       ;            i.e force an exit.
14992                                  <1>       ;         - if arg is an addres in the program,
14993                                  <1>       ;           a quit causes control to sent to that
```

```
14994                                  <1>       ;            location.
14995                                  <1>       ; Inputs: -
14996                                  <1>       ; Outputs: -
14997                                  <1>       ; ..........................................................
14998                                  <1>       ;
14999                                  <1>       ; Retro UNIX 8086 v1 modification:
15000                                  <1>       ;       'sysquit' system call sets u.quit to value of BX
15001                                  <1>       ;     then branches into 'sysret'.
15002                                  <1>       ;
15003 00011099 66891D[AC030300]        <1>       mov    [u.quit], bx
15004 000110A0 E927C7FFFF              <1>       jmp    sysret
15005                                  <1>              ; jsr r0,arg; u.quit / put argument in u.quit
15006                                  <1>       ;1:
15007                                  <1>              ; mov u.ttyp,r1 / move pointer to control tty buffer
15008                                  <1>                              ; / to r1
15009                                  <1>              ; beq sysret4 / return to user
15010                                  <1>              ; clrb 6(r1) / clear the interrupt character
15011                                  <1>                       ; / in the tty buffer
15012                                  <1>              ; br sysret4 / return to user
15013                                  <1>
15014                                  <1> anyi:
15015                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
15016                                  <1>       ; Major Modification!
15017                                  <1>       ; TRDOS 386 does not permit to delete a file while it is open
15018                                  <1>       ; The role of 'anyi' procedure has beeen changed to ensure that.
15019                                  <1>       ;
15020                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
15021                                  <1>       ; 25/04/2013 (Retro UNIX 8086 v1)
15022                                  <1>       ;
15023                                  <1>       ; 'anyi' is called if a file deleted while open.
15024                                  <1>       ; "anyi" checks to see if someone else has opened this file.
15025                                  <1>       ;
15026                                  <1>       ; INPUTS ->
15027                                  <1>       ;    r1 - contains an i-number
15028                                  <1>       ;    fsp - start of table containing open files
15029                                  <1>       ;
15030                                  <1>       ; OUTPUTS ->
15031                                  <1>       ;    "deleted" flag set in fsp entry of another occurrence of
15032                                  <1>       ;        this file and r2 points 1st word of this fsp entry.
15033                                  <1>       ;    if file not found - bit in i-node map is cleared
15034                                  <1>       ;                     (i-node is freed)
15035                                  <1>       ;              all blocks related to i-node are freed
15036                                  <1>       ;             all flags in i-node are cleared
15037                                  <1>       ; ((AX = R1)) input
15038                                  <1>       ;
15039                                  <1>       ;    (Retro UNIX Prototype : 02/12/2012, UNIXCOPY.ASM)
15040                                  <1>         ;    ((Modified registers: eDX, eCX, eBX, eSI, eDI, eBP))
15041                                  <1>       ;
15042                                  <1>       ; / r1 contains an i-number
15043                                  <1>
15044                                  <1>       ; TRDOS 386 (06/10/2016)
15045                                  <1>       ;
15046                                  <1>       ; INPUT:
15047                                  <1>       ;     EAX = First Cluster
15048                                  <1>       ;      DL = Logical DOS Drive Number
15049                                  <1>       ;
15050                                  <1>       ; OUTPUT:
15051                                  <1>       ;     CF = 1 -> EBX = File Handle/Number/Index
15052                                  <1>       ;     CF = 0 -> EBX = 0
15053                                  <1>       ;
15054                                  <1>       ; Modified Registers: EBX
15055                                  <1>
15056 000110A5 31DB                   <1>       xor    ebx, ebx
15057                                  <1> anyi_0:
15058 000110A7 80BB[16990100]00        <1>       cmp    byte [ebx+OF_MODE], 0 ; 0 = empty entry
15059 000110AE 770A                   <1>       ja     short anyi_2 ; 1 (r), 2 (w) or 3 (r&w)
15060                                  <1> anyi_1:
15061 000110B0 FEC3                   <1>       inc    bl
15062 000110B2 80FB0A                 <1>       cmp    bl, OPENFILES ; max. count of open files
15063 000110B5 72F0                   <1>       jb     short anyi_0
15064 000110B7 31C0                   <1>       xor    eax, eax
15065 000110B9 C3                     <1>       retn
15066                                  <1> anyi_2:
15067 000110BA 3A93[0C990100]         <1>       cmp    dl, [ebx+OF_DRIVE]
15068 000110C0 75EE                   <1>       jne    short anyi_1
15069 000110C2 66C1E302               <1>       shl    bx, 2 ; *4 (dword offset)
15070 000110C6 3B83[E4980100]         <1>       cmp    eax, [ebx+OF_FCLUSTER]
15071 000110CC 7406                   <1>       je     short anyi_3
15072 000110CE 66C1EB02               <1>       shr    bx, 2 ; /4 (byte offset)
15073 000110D2 EBDC                   <1>       jmp    short anyi_1
15074                                  <1> anyi_3:
15075 000110D4 66C1EB02               <1>       shr    bx, 2 ; /4 (bytes offset) (index)
15076 000110D8 F9                     <1>       stc
15077 000110D9 C3                     <1>       retn
15078                                  <1>
15079                                  <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS9.INC
15080                                  <1> ; Last Modification: 09/12/2015
15081                                  <1>
15082                                  <1> syssleep:
15083                                  <1>       ; 29/06/2015 - (Retro UNIX 386 v1)
15084                                  <1>       ; 11/06/2014 - (Retro UNIX 8086 v1)
15085                                  <1>       ;
15086                                  <1>       ; Retro UNIX 8086 v1 feature only
15087                                  <1>       ; (INPUT -> none)
15088                                  <1>       ;
15089 000110DA 0FB61D[B3030300]       <1>       movzx  ebx, byte [u.uno] ; process number
15090 000110E1 8AA3[7F000300]         <1>       mov    ah, [ebx+p.ttyc-1] ; current/console tty
15091 000110E7 E881190000             <1>       call   sleep
15092 000110EC E9DBC6FFFF             <1>       jmp    sysret
15093                                  <1>
15094                                  <1> _vp_clr:
15095                                  <1>       ; Reset/Clear Video Page
15096                                  <1>       ;
15097                                  <1>       ; 30/06/2015 - (Retro UNIX 386 v1)
15098                                  <1>       ; 21/05/2013 - 30/10/2013(Retro UNIX 8086 v1) (U0.ASM)
```

```
15099                                  <1>       ;
15100                                  <1>       ; Retro UNIX 8086 v1 feature only !
15101                                  <1>       ;
15102                                  <1>       ; INPUTS ->
15103                                  <1>       ;   BH = video page number
15104                                  <1>       ;
15105                                  <1>       ; OUTPUT ->
15106                                  <1>       ;   none
15107                                  <1>       ; ((Modified registers: eAX, BH, eCX, eDX, eSI, eDI))
15108                                  <1>       ;
15109                                  <1>       ; 04/12/2013
15110 000110F1 28C0                   <1>       sub    al, al
15111                                  <1>       ; al = 0 (clear video page)
15112                                  <1>       ; bh = video page ; 13/05/2016
15113 000110F3 B407                   <1>       mov    ah, 07h
15114                                  <1>       ; ah = 7 (attribute/color)
15115 000110F5 6631C9                 <1>       xor    cx, cx ; 0, left upper column (cl) & row (cl)
15116 000110F8 66BA4F18               <1>       mov    dx, 184Fh ; right lower column & row (dl=24, dh=79)
15117 000110FC E8D20EFFFF             <1>       call   _scroll_up
15118                                  <1>       ; bh = video page
15119 00011101 6631D2                 <1>       xor    dx, dx ; 0 (cursor position)
15120 00011104 E91112FFFF             <1>       jmp    _set_cpos
15121                                  <1>
15122                                  <1> sysmsg:
15123                                  <1>       ; 07/12/2020
15124                                  <1>       ; 05/12/2020
15125                                  <1>       ; 13/05/2016
15126                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
15127                                  <1>       ; 01/07/2015 - 11/11/2015 (Retro UNIX 386 v1)
15128                                  <1>       ; Print user-application message on user's console tty
15129                                  <1>       ;
15130                                  <1>       ; Input -> EBX = Message address
15131                                  <1>       ;          ECX = Message length (max. 255)
15132                                  <1>       ;          DL = Color (IBM PC Rombios color attributes)
15133                                  <1>       ;
15134 00011109 81F9FF000000           <1>       cmp    ecx, MAX_MSG_LEN ; 255
15135 0001110F 0F87B7C6FFFF           <1>       ja     sysret ; nothing to do with big message size
15136 00011115 08C9                   <1>       or     cl, cl
15137 00011117 0F84AFC6FFFF           <1>       jz     sysret
15138 0001111D 20D2                   <1>       and    dl, dl
15139 0001111F 7502                   <1>       jnz    short sysmsg0
15140 00011121 B207                   <1>       mov    dl, 07h ; default color
15141                                  <1>              ; (black background, light gray character)
15142                                  <1> sysmsg0:
15143 00011123 891D[84030300]         <1>       mov    [u.base], ebx
15144 00011129 8815[1F890100]         <1>       mov    [ccolor], dl ; color attributes
15145 0001112F 89E5                   <1>       mov    ebp, esp
15146 00011131 31DB                   <1>       xor    ebx, ebx ; 0
15147 00011133 891D[8C030300]         <1>       mov    [u.nread], ebx ; 0
15148                                  <1>       ;
15149 00011139 381D[C6030300]         <1>       cmp    [u.kcall], bl ; 0
15150 0001113F 776F                   <1>       ja     short sysmsgk ; Temporary (01/07/2015)
15151                                  <1>       ;
15152 00011141 890D[88030300]         <1>       mov    [u.count], ecx
15153                                  <1>       ;inc   ecx ; + 00h ; ASCIIZ
15154                                  <1>       ;
15155                                  <1>       ; 07/12/2020
15156                                  <1>       ;add   ecx, 3
15157 00011147 6683C103               <1>       add    cx, 3
15158 0001114B 80E1FC                 <1>       and    cl, ~3  ; not 3
15159                                  <1>       ;
15160 0001114E 29CC                   <1>       sub    esp, ecx
15161 00011150 89E7                   <1>       mov    edi, esp
15162 00011152 89E6                   <1>       mov    esi, esp
15163 00011154 66891D[C4030300]       <1>       mov    [u.pcount], bx ; reset page (phy. addr.) counter
15164                                  <1>       ; 11/11/2015
15165 0001115B 8A25[94030300]         <1>       mov    ah, [u.ttyp] ; recent open tty
15166                                  <1>       ; 0 = none
15167 00011161 FECC                   <1>       dec    ah
15168 00011163 790C                   <1>       jns    short sysmsg1
15169 00011165 8A1D[B3030300]         <1>       mov    bl, [u.uno] ; process number
15170 0001116B 8AA3[7F000300]         <1>       mov    ah, [ebx+p.ttyc-1] ; user's (process's) console tty
15171                                  <1> sysmsg1:
15172 00011171 8825[96030300]         <1>       mov    [u.ttyn], ah
15173                                  <1> sysmsg2:
15174 00011177 E848080000             <1>       call   cpass
15175 0001117C 7416                   <1>       jz     short sysmsg5
15176 0001117E AA                     <1>       stosb
15177 0001117F 20C0                   <1>       and    al, al
15178 00011181 75F4                   <1>       jnz    short sysmsg2
15179                                  <1> sysmsg3:
15180 00011183 80FC07                 <1>       cmp    ah, 7 ; tty number
15181 00011186 7711                   <1>       ja     short sysmsg6 ; serial port
15182 00011188 E83E000000             <1>       call   print_cmsg ; 05/12/2020
15183                                  <1> sysmsg4:
15184 0001118D 89EC                   <1>       mov    esp, ebp
15185 0001118F E938C6FFFF             <1>       jmp    sysret
15186                                  <1> sysmsg5:
15187 00011194 C60700                 <1>       mov    byte [edi], 0
15188 00011197 EBEA                   <1>       jmp    short sysmsg3
15189                                  <1> sysmsg6:
15190 00011199 8A06                   <1>       mov    al, [esi]
15191 0001119B E8CD180000             <1>       call   sndc
15192 000111A0 72EB                   <1>       jc     short sysmsg4
15193 000111A2 803E00                 <1>       cmp    byte [esi], 0  ; 0 is stop character
15194 000111A5 76E6                   <1>       jna    short sysmsg4
15195 000111A7 46                     <1>       inc    esi
15196 000111A8 8A25[96030300]         <1>       mov    ah, [u.ttyn]
15197 000111AE EBE9                   <1>       jmp    short sysmsg6
15198                                  <1>
15199                                  <1> sysmsgk: ; Temporary (01/07/2015)
15200                                  <1>       ; The message has been sent by Kernel (ASCIIZ string)
15201                                  <1>       ; (ECX -character count- will not be considered)
15202 000111B0 8B35[84030300]         <1>       mov    esi, [u.base]
15203 000111B6 8A25[1E890100]         <1>       mov    ah, [ptty] ; present/current screen (video page)
```

```
15204 000111BC 8825[96030300]      <1>       mov    [u.ttyn], ah
15205 000111C2 C605[C6030300]00    <1>       mov    byte [u.kcall], 0
15206 000111C9 EBB8                <1>       jmp    short sysmsg3
15207                              <1>
15208                              <1> print_cmsg:
15209                              <1>       ; 08/12/2020
15210                              <1>       ; 07/12/2020
15211                              <1>       ; 05/12/2020
15212                              <1>       ; 18/11/2017
15213                              <1>       ; 13/05/2016 - TRDOS 386 (TRDOS v2.0)
15214                              <1>       ; 01/07/2015 (Retro UNIX 386 v1)
15215                              <1>       ;
15216                              <1>       ; print message (on user's console tty)
15217                              <1>       ;      with requested color
15218                              <1>       ;
15219                              <1>       ; INPUTS:
15220                              <1>       ;      esi = message address
15221                              <1>       ;      [u.ttyn] = tty number (0 to 7)
15222                              <1>       ;      [ccolor] = color attributes (IBM PC BIOS colors)
15223                              <1>       ;
15224                              <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
15225                              <1>       ; (ebp must be preserved)
15226                              <1>
15227                              <1>       ;mov   bh, ah
15228 000111CB 8A3D[96030300]      <1>       mov    bh, [u.ttyn]
15229 000111D1 8A1D[1F890100]      <1>       mov    bl, [ccolor] ; * ; 05/12/2020
15230                              <1>
15231                              <1>       ; 05/12/2020
15232 000111D7 803D[1C120300]00    <1>       cmp    byte [pmi32], 0 ; is vbios's 32 bit pmi enabled ?
15233 000111DE 772E                <1>       ja     short pcmsg5 ; yes
15234                              <1> pcmsg1:
15235                              <1>       ; 08/12/2020
15236 000111E0 8A1D[1F890100]      <1>       mov    bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
15237                              <1>
15238 000111E6 AC                  <1>       lodsb
15239 000111E7 20C0                <1>       and    al, al  ; 0
15240 000111E9 743A                <1>       jz     short pcmsg2
15241                              <1> pcmsg7:
15242 000111EB 56                  <1>       push   esi
15243                              <1>       ;mov   bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
15244                              <1>       ; 05/12/2020
15245                              <1>       ;;mov  bh, [u.ttyn]
15246                              <1>       ;call  _write_tty
15247                              <1>       ;pop   esi
15248                              <1>       ;jmp   short pcmsg1
15249                              <1> ;pcmsg2:
15250                              <1>       ;retn
15251                              <1>
15252                              <1>       ; 07/12/2020
15253 000111EC 803D[9A6E0000]03    <1>       cmp     byte [CRT_MODE], 3
15254 000111F3 7708                <1>       ja     short pcmsg4
15255                              <1> pcmsg3:
15256 000111F5 E89B10FFFF          <1>       call   _write_tty_m3
15257 000111FA 5E                  <1>       pop    esi
15258 000111FB EBE3                <1>       jmp    short pcmsg1
15259                              <1> pcmsg4:
15260 000111FD 803D[9A6E0000]07    <1>         cmp     byte [CRT_MODE], 7
15261 00011204 76EF                <1>       jna    short pcmsg3
15262 00011206 E89A1DFFFF          <1>       call   vga_write_teletype
15263 0001120B 5E                  <1>       pop    esi
15264 0001120C EBD2                <1>       jmp    short pcmsg1
15265                              <1> pcmsg5:
15266                              <1>       ; 07/12/2020
15267 0001120E 803D[9A6E0000]07    <1>         cmp     byte [CRT_MODE], 7
15268 00011215 76C9                <1>       jna    short pcmsg1
15269                              <1>
15270                              <1>       ; 05/12/2020
15271                              <1>       ; writing message by using
15272                              <1>       ; VESA VBE3 video bios protected mode interface
15273                              <1>
15274 00011217 B40E                <1>       mov    ah, 0Eh
15275                              <1> pcmsg6:
15276 00011219 AC                  <1>       lodsb
15277 0001121A 20C0                <1>       and    al, al  ; 0
15278 0001121C 7407                <1>       jz     short pcmsg2
15279                              <1>       ; bh = video page
15280                              <1>       ; ah = 0Eh
15281                              <1>       ; al = character
15282                              <1>       ; bl = color
15283 0001121E E88107FFFF          <1>       call   int10h_32bit_pmi
15284 00011223 EBF4                <1>       jmp    short pcmsg6
15285                              <1> pcmsg2:
15286 00011225 C3                  <1>       retn
15287                              <1>
15288                              <1> sysgeterr:
15289                              <1>       ; 09/12/2015
15290                              <1>       ; 21/09/2015 - (Retro UNIX 386 v1 feature only!)
15291                              <1>       ; Get last error number or page fault count
15292                              <1>       ; (for debugging)
15293                              <1>       ;
15294                              <1>       ; Input -> EBX = return type
15295                              <1>       ;         0 = last error code (which is in 'u.error')
15296                              <1>       ;         FFFFFFFFh = page fault count for running process
15297                              <1>       ;         FFFFFFFEh = total page fault count
15298                              <1>       ;         1 .. FFFFFFFDh = undefined
15299                              <1>       ;
15300                              <1>       ; Output -> EAX = last error number or page fault count
15301                              <1>       ;         (depending on EBX input)
15302                              <1>       ;
15303 00011226 21DB                <1>       and    ebx, ebx
15304 00011228 750B                <1>       jnz    short glerr_2
15305                              <1> glerr_0:
15306 0001122A A1[C8030300]        <1>       mov    eax, [u.error]
15307                              <1> glerr_1:
15308 0001122F A3[64030300]        <1>       mov    [u.r0], eax
```

```
15309 00011234 C3                <1>       retn
15310                            <1> glerr_2:
15311 00011235 43                <1>       inc    ebx ; FFFFFFFFh -> 0, FFFFFFFEh -> FFFFFFFFh
15312 00011236 74FD              <1>       jz     short glerr_2 ; page fault count for process
15313 00011238 43                <1>       inc    ebx ; FFFFFFFFh -> 0
15314 00011239 75EF              <1>       jnz    short glerr_0
15315 0001123B A1[80050300]      <1>       mov    eax, [PF_Count] ; total page fault count
15316 00011240 EBED              <1>         jmp     short glerr_1
15317                            <1> glerr_3:
15318 00011242 A1[CC030300]      <1>       mov    eax, [u.pfcount]
15319 00011247 EBE6              <1>       jmp    short glerr_1
15320                            <1>
15321                            <1> load_and_run_file:
15322                            <1>       ; 18/11/2017
15323                            <1>       ; 22/01/2017
15324                            <1>       ; 04/01/2017, 07/01/2017
15325                            <1>       ; 24/10/2016
15326                            <1>       ; 24/04/2016, 02/05/2016, 03/05/2016, 06/05/2016
15327                            <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
15328                            <1>       ; 23/10/2015 (Retro UNIX 386 v1, 'sysexec')
15329                            <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
15330                            <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
15331                            <1>       ; EAX = First Cluster number
15332                            <1>       ; EDX = File Size
15333                            <1>       ; ESI = Argument list address
15334                            <1>       ; [argc] = argument count
15335                            <1>       ; [u.nread] = argument list length
15336                            <1>       ; [esp] = return address to the caller (*)
15337                            <1>       ;
15338 00011249 8935[4C040300]    <1>       mov    [argv], esi
15339 0001124F 8915[55040300]    <1>       mov    [i.size], edx
15340 00011255 A3[51040300]      <1>       mov    [ii], eax
15341                            <1>
15342                            <1>       ;sti   ; 07/01/2017
15343                            <1>       ;mov   eax, [k_page_dir]
15344                            <1>       ;mov   [u.pgdir], eax
15345 0001125A 31C0              <1>       xor    eax, eax ; clc ; *** ; 04/01/2017
15346                            <1>       ;mov   [u.r0], eax ; 0 ; 07/01/2017
15347                            <1>
15348                            <1>       ; 06/05/2016
15349                            <1>       ; Set 'sysexit' return order to MainProg
15350                            <1>       ;
15351 0001125C 58                <1>       pop    eax ; * 'loc_load_and_run_file_8:' address
15352                            <1>       ;; 22/01/2017
15353                            <1>       ;;cli ; 07/01/2017
15354 0001125D 8B25[8C880100]    <1>       mov    esp, [tss.esp0]
15355                            <1>       ;
15356                            <1>       ; 'loc_load_run_file_8' address has
15357                            <1>       ; 'jmp loc_file_rw_restore_retn' instruction
15358                            <1>       ; 'loc_file_rw_restore_retn:' will return to
15359                            <1>       ; [mainprog_return_addr]
15360                            <1>       ; just after 'call command_interpreter'
15361                            <1>       ;
15362 00011263 68[03740000]      <1>       push   _end_of_mainprog ; we must not return to here !
15363 00011268 FF35[70950100]    <1>       push   dword [mainprog_return_addr]
15364 0001126E 89E5              <1>       mov    ebp, esp ; **
15365                            <1>       ;
15366 00011270 9C                <1>       pushfd  ; EFLAGS      ; IRETD ; ***
15367 00011271 6A08              <1>       push   KCODE ; cs    ; IRETD
15368 00011273 50                <1>       push   eax ; * (eip) ; IRETD
15369 00011274 8925[5C030300]    <1>       mov    [u.sp], esp
15370                            <1>       ;mov   byte [u.quant], time_count
15371 0001127A 1E                <1>       push   ds
15372 0001127B 06                <1>       push   es
15373 0001127C 0FA0              <1>       push   fs
15374 0001127E 0FA8              <1>       push   gs
15375                            <1>       ;mov   eax, [u.r0]
15376 00011280 29C0              <1>       sub    eax, eax
15377 00011282 60                <1>       pushad
15378 00011283 68[CCD70000]      <1>       push   sysret
15379                            <1>       ;push  sysrel1 ; 07/01/2017
15380 00011288 8925[60030300]    <1>       mov    [u.usp], esp
15381                            <1>       ;
15382 0001128E E845060000        <1>       call   wswap ; Save MainProg (process 1) 'u' structure
15383                            <1>                    ; and registers for return (from program)
15384 00011293 89EC              <1>       mov    esp, ebp ; **
15385                            <1>       ;;22/01/2017
15386                            <1>       ;;sti ; 07/01/2017
15387 00011295 50                <1>       push   eax  ; * 'loc_load_and_run_file_8:' address
15388                            <1>       ;
15389                            <1>       ;;; 02/05/2016
15390                            <1>       ;;; Create a new process (parent: MainProg)
15391 00011296 31F6              <1>       xor    esi, esi
15392                            <1> cnpm_1: ; search p.stat table for unused process number
15393 00011298 46                <1>       inc    esi
15394 00011299 80BE[AF000300]00  <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE
15395                            <1>                           ; is process active, unused, dead
15396 000112A0 760B              <1>       jna    short cnpm_2 ; it's unused so branch
15397 000112A2 6683FE10          <1>       cmp    si, nproc    ; all processes checked
15398 000112A6 72F0              <1>       jb     short cnpm_1    ; no, branch back
15399 000112A8 E9DE61FFFF        <1>       jmp    panic
15400                            <1> cnpm_2:
15401 000112AD A1[B8030300]      <1>       mov    eax, [u.pgdir] ; page directory of MainProg
15402 000112B2 A3[BC030300]      <1>       mov    [u.ppgdir], eax ; parent's page directory
15403 000112B7 E8EB47FFFF        <1>       call   allocate_page
15404 000112BC 0F82C961FFFF      <1>       jc     panic
15405                            <1>       ; EAX = UPAGE (user structure page) address
15406 000112C2 A3[B4030300]      <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
15407 000112C7 89F7              <1>       mov    edi, esi
15408 000112C9 66C1E702          <1>       shl    di, 2
15409 000112CD 8987[BC000300]    <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
15410 000112D3 E84948FFFF        <1>       call   clear_page ; 03/05/2016
15411                            <1>       ;movzx eax, byte [p.ttyc] ; console tty (for MainProg)
15412 000112D8 6629C0            <1>       sub    ax, ax ; 0
15413 000112DB 668986[7F000300]  <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
```

```
15414                                  <1>                             ; ah - reset child's wait channel
15415 000112E2 89F0                    <1>       mov   eax, esi
15416 000112E4 A2[B3030300]            <1>       mov   [u.uno], al ; child process number
15417 000112E9 FE86[AF000300]          <1>         inc     byte [esi+p.stat-1] ; 1, SRUN
15418 000112EF 66D1E6                  <1>       shl   si, 1 ; multiply si by 2 to get index into p.pid table
15419 000112F2 66FF05[4E030300]        <1>       inc   word [mpid] ; increment m.pid; get a new process name
15420 000112F9 66A1[4E030300]          <1>       mov   ax, [mpid]
15421 000112FF 668986[1E000300]        <1>       mov   [esi+p.pid-2], ax ; put new process name
15422                                  <1>                             ; in child process' name slot
15423                                  <1>       ;mov  ax, [p.pid]  ; get process name of MainProg
15424 00011306 66B80100                <1>       mov   ax, 1
15425 0001130A 668986[3E000300]        <1>       mov   [esi+p.ppid-2], ax ; put parent process name
15426                                  <1>                               ; in parent process slot for child
15427 00011311 6648                    <1>       dec   ax ; 0
15428 00011313 66A3[94030300]          <1>       mov   [u.ttyp], ax ; 0
15429                                  <1>       ;;;
15430 00011319 A1[51040300]            <1>       mov   eax,  [ii]
15431                                  <1>       ; Retro UNIX 386 v1, 'sysexec' (u2.s)
15432 0001131E E84A170000              <1>       call  iopen
15433                                  <1>       ; 06/06/2016
15434 00011323 C605[A9030300]01        <1>       mov   byte [u.pri], 1 ; normal priority
15435                                  <1>       ;
15436 0001132A EB16                    <1>       jmp   short sysexec_7 ; 02/05/2016
15437                                  <1>
15438                                  <1> sysexec_6:
15439                                  <1>       ; 19/11/2017
15440                                  <1>       ; 18/11/2017
15441                                  <1>       ; 14/11/2017
15442                                  <1>       ; 13/11/2017
15443 0001132C 8925[4C040300]          <1>       mov   [argv], esp ; *!* ; start address of argument list
15444                                  <1>
15445                                  <1>       ; 04/01/2017
15446                                  <1>       ; 24/10/2016
15447                                  <1>       ;;02/05/2016
15448                                  <1>       ; 23/04/2016 (TRDOS 386)
15449                                  <1>       ; 18/10/2015 ('sysexec_6')
15450                                  <1>       ; 23/06/2015
15451 00011332 A1[B8030300]            <1>       mov   eax, [u.pgdir] ; physical address of page directory
15452                                  <1>       ;cmp  eax, [k_page_dir] ; TRDOS MainProg ?
15453                                  <1>       ;je   short sysexec_7
15454                                  <1>       ; 19/11/2017
15455 00011337 8B1D[BC030300]          <1>       mov   ebx, [u.ppgdir] ; phy addr of the parent's page dir
15456 0001133D E89E48FFFF              <1>       call  deallocate_page_dir
15457                                  <1> sysexec_7:
15458 00011342 E8CE47FFFF              <1>       call  make_page_dir
15459 00011347 0F823E61FFFF            <1>       jc    panic  ; allocation error
15460                                  <1>                    ; after a deallocation would be nonsence !?
15461                                  <1>       ; 24/07/2015
15462                                  <1>       ; map kernel pages (1st 4MB) to PDE 0
15463                                  <1>       ;     of the user's page directory
15464                                  <1>       ;     (It is needed for interrupts!)
15465                                  <1>       ; 18/10/2015
15466 0001134D 8B15[F0880100]          <1>       mov   edx, [k_page_dir] ; Kernel's page directory
15467 00011353 8B02                    <1>       mov   eax, [edx] ; physical address of
15468                                  <1>                        ; kernel's first page table (1st 4 MB)
15469                                  <1>                        ; (PDE 0 of kernel's page directory)
15470 00011355 8B15[B8030300]          <1>       mov   edx, [u.pgdir]
15471 0001135B 8902                    <1>       mov   [edx], eax ; PDE 0 (1st 4MB)
15472                                  <1>       ;
15473                                  <1>       ; 20/07/2015
15474 0001135D BB00004000              <1>       mov   ebx, CORE ; start address = 0 (virtual) + CORE
15475                                  <1>       ; 18/10/2015
15476 00011362 BE[3C040300]            <1>       mov   esi, pcore ; physical start address
15477                                  <1> sysexec_8:
15478 00011367 B907000000              <1>       mov   ecx, PDE_A_USER + PDE_A_WRITE + PDE_A_PRESENT
15479 0001136C E8C247FFFF              <1>       call  make_page_table
15480 00011371 0F821461FFFF            <1>       jc    panic
15481                                  <1>       ;mov  ecx, PTE_A_USER + PTE_A_WRITE + PTE_A_PRESENT
15482 00011377 E8C547FFFF              <1>       call  make_page ; make new page, clear and set the pte
15483 0001137C 0F820961FFFF            <1>       jc    panic
15484                                  <1>       ;
15485 00011382 8906                    <1>       mov   [esi], eax ; 24/06/2015
15486                                  <1>       ; ebx = virtual address (24/07/2015)
15487 00011384 E85D4DFFFF              <1>       call  add_to_swap_queue
15488                                  <1>       ; 18/10/2015
15489 00011389 81FE[40040300]          <1>       cmp   esi, ecore ; user's stack (last) page ?
15490 0001138F 740C                    <1>       je    short sysexec_9 ; yes
15491 00011391 BE[40040300]            <1>       mov   esi, ecore  ; physical address of the last page
15492                                  <1>       ; 20/07/2015
15493 00011396 BB00F0FFFF              <1>       mov   ebx, (ECORE - PAGE_SIZE) + CORE
15494                                  <1>       ; ebx = virtual end address + segment base address - 4K
15495 0001139B EBCA                    <1>         jmp     short sysexec_8
15496                                  <1> sysexec_9:
15497                                  <1>       ; 19/11/2017
15498                                  <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
15499                                  <1>       ; 25/06/2015, 26/08/2015, 18/10/2015
15500                                  <1>       ; move arguments from kernel stack to [ecore]
15501                                  <1>       ; (argument list/line will be copied from kernel stack
15502                                  <1>       ; frame to the last (stack) page of user's core memory)
15503                                  <1>       ; 18/10/2015
15504 0001139D 8B3D[40040300]          <1>       mov   edi, [ecore]
15505 000113A3 81C700100000            <1>       add   edi, PAGE_SIZE
15506                                  <1>       ; 19/11/2017
15507 000113A9 83EF04                  <1>       sub   edi, 4
15508 000113AC C70700000000            <1>       mov   dword [edi], 0
15509 000113B2 89FB                    <1>       mov   ebx, edi
15510                                  <1>       ;
15511 000113B4 0FB705[4A040300]        <1>       movzx  eax, word [argc]
15512 000113BB 09C0                    <1>       or    eax, eax
15513 000113BD 7445                    <1>       jz    short sysexec_13 ; 19/11/2017
15514                                  <1>       ;jnz  short sysexec_10
15515                                  <1>       ;mov  ebx, edi
15516                                  <1>       ;sub  ebx, 4
15517                                  <1>       ;mov  [ebx], eax ; 0
15518                                  <1>       ;jmp  short sysexec_13
```

```
15519                                  <1> sysexec_10:
15520 000113BF 8B0D[8C030300]          <1>       mov   ecx, [u.nread]
15521                                  <1>       ; 13/11/2017
15522                                  <1>       ;mov  esi, TextBuffer ; 'load_and_execute_file'
15523                                  <1>       ;mov  esi, esp     ; 'sysexec'
15524 000113C5 8B35[4C040300]          <1>       mov   esi, [argv] ; 24/04/2016 (TRDOS 386  = TRDOS v2.0)
15525                                  <1>       ;sub  edi, ecx ; page end address - argument list length
15526 000113CB 29CB                    <1>       sub   ebx, ecx ; 19/11/2017
15527 000113CD 89C2                    <1>       mov   edx, eax
15528 000113CF FEC2                    <1>       inc   dl ; argument count + 1 for argc value
15529 000113D1 C0E202                  <1>       shl   dl, 2  ; 4 * (argument count + 1)
15530                                  <1>       ;mov  ebx, edi
15531 000113D4 89DF                    <1>       mov   edi, ebx ; 19//11/2017
15532 000113D6 80E3FC                  <1>       and   bl, 0FCh ; 32 bit (dword) alignment
15533 000113D9 29D3                    <1>       sub   ebx, edx
15534 000113DB 89FA                    <1>       mov   edx, edi
15535 000113DD F3A4                    <1>       rep   movsb
15536 000113DF 89D6                    <1>       mov   esi, edx
15537 000113E1 89DF                    <1>       mov   edi, ebx
15538 000113E3 BA00F0BFFF              <1>       mov   edx, ECORE - PAGE_SIZE ; virtual addr. of the last page
15539 000113E8 2B15[40040300]          <1>       sub   edx, [ecore] ; difference (virtual - physical)
15540 000113EE AB                      <1>       stosd ; eax = argument count
15541                                  <1> sysexec_11:
15542 000113EF 89F0                    <1>       mov   eax, esi
15543 000113F1 01D0                    <1>       add   eax, edx
15544 000113F3 AB                      <1>       stosd  ; eax = virtual address
15545                                  <1>       ;dec  byte [argc]
15546 000113F4 66FF0D[4A040300]        <1>       dec   word [argc] ; 14/11/2017
15547 000113FB 7407                    <1>       jz    short sysexec_13
15548                                  <1> sysexec_12:
15549 000113FD AC                      <1>       lodsb
15550 000113FE 20C0                    <1>       and   al, al
15551 00011400 75FB                    <1>       jnz   short sysexec_12
15552 00011402 EBEB                    <1>       jmp   short sysexec_11
15553                                  <1> sysexec_13:
15554                                  <1>       ; 24/10/2016
15555                                  <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
15556                                  <1>       ; 23/06/2015 - 19/10/2015 (Retro UNIX 386 v1, 'sysexec_13')
15557                                  <1>       ;
15558                                  <1>       ; moving arguments to [ecore] is OK here..
15559                                  <1>       ;
15560                                  <1>       ; ebx = beginning addres of argument list pointers
15561                                  <1>             ;     in user's stack
15562 00011404 2B1D[40040300]          <1>       sub   ebx, [ecore]
15563 0001140A 81C300F0BFFF            <1>       add    ebx, (ECORE - PAGE_SIZE)
15564                                  <1>                  ; end of core - 4096 (last page)
15565                                  <1>                  ; (virtual address)
15566 00011410 891D[4C040300]          <1>       mov   [argv], ebx
15567 00011416 891D[90030300]          <1>       mov   [u.break], ebx ; available user memory
15568                                  <1>       ;
15569 0001141C 29C0                    <1>       sub   eax, eax
15570 0001141E C705[88030300]2000-     <1>       mov   dword [u.count], 32 ; Executable file header size
15570 00011426 0000                    <1>
15571 00011428 C705[74030300]-         <1>       mov   dword [u.fofp], u.off
15571 0001142E [80030300]              <1>
15572 00011432 A3[80030300]            <1>       mov   [u.off], eax ; 0
15573 00011437 A3[84030300]            <1>       mov   [u.base], eax ; 0, start of user's core (virtual)
15574                                  <1>       ; 24/10/2016
15575 0001143C A0[B6890100]            <1>       mov   al, [Current_Drv]
15576 00011441 A2[46030300]            <1>       mov   [cdev], al
15577                                  <1>       ;
15578 00011446 A1[51040300]            <1>       mov   eax, [ii] ; Fist Cluster of the Program (PRG) file
15579                                  <1>       ; EAX = First cluster of the executable file
15580 0001144B E80A010000              <1>       call  readi
15581                                  <1>
15582 00011450 8B0D[90030300]          <1>       mov   ecx, [u.break] ; top of user's stack (physical addr.)
15583 00011456 890D[88030300]          <1>       mov   [u.count], ecx ; save for overrun check
15584                                  <1>       ;
15585 0001145C 8B0D[8C030300]          <1>       mov   ecx, [u.nread]
15586 00011462 890D[90030300]          <1>       mov   [u.break], ecx ; virtual address (offset from start)
15587 00011468 80F920                  <1>       cmp   cl, 32
15588 0001146B 7540                    <1>         jne     short sysexec_15
15589                                  <1>       ;:
15590                                  <1>       ; Retro UNIX 386 v1 (32 bit) executable file header format
15591 0001146D 8B35[3C040300]          <1>       mov   esi, [pcore] ; start address of user's core memory
15592                                  <1>                          ; (phys. start addr. of the exec. file)
15593 00011473 AD                      <1>       lodsd
15594 00011474 663DEB1E                <1>       cmp   ax, 1EEBh ; EBH, 1Eh -> jump to +32
15595 00011478 7533                    <1>       jne   short sysexec_15
15596 0001147A AD                      <1>       lodsd
15597 0001147B 89C1                    <1>       mov   ecx, eax ; text (code) section size
15598 0001147D AD                      <1>       lodsd
15599 0001147E 01C1                    <1>       add   ecx, eax ; + data section size (initialized data)
15600 00011480 89CB                    <1>       mov   ebx, ecx
15601 00011482 AD                      <1>       lodsd
15602 00011483 01C3                    <1>       add   ebx, eax ; + bss section size (for overrun checking)
15603 00011485 3B1D[88030300]          <1>       cmp   ebx, [u.count]
15604 0001148B 7711                    <1>       ja    short sysexec_14  ; program overruns stack !
15605                                  <1>       ;
15606                                  <1>       ; add bss section size to [u.break]
15607 0001148D 0105[90030300]          <1>       add   [u.break], eax
15608                                  <1>       ;
15609 00011493 83E920                  <1>       sub   ecx, 32  ; header size (already loaded)
15610                                  <1>       ;cmp  ecx, [u.count]
15611                                  <1>       ;jnb  short sysexec_16
15612 00011496 890D[88030300]          <1>       mov   [u.count], ecx ; required read count
15613 0001149C EB29                    <1>       jmp   short sysexec_16
15614                                  <1> sysexec_14:
15615                                  <1>       ; insufficient (out of) memory
15616 0001149E C705[C8030300]0400-     <1>       mov   dword [u.error], ERR_MINOR_IM ; 1
15616 000114A6 0000                    <1>
15617 000114A8 E9FFC2FFFF              <1>       jmp   error
15618                                  <1> sysexec_15:
15619 000114AD 8B15[55040300]          <1>         mov  edx, [i.size] ; file size
15620 000114B3 29CA                    <1>       sub   edx, ecx ; file size - loaded bytes
```

```
15621 000114B5 7626              <1>       jna    short sysexec_17 ; no need to next read
15622 000114B7 01D1              <1>       add    ecx, edx ; [i.size]
15623 000114B9 3B0D[88030300]    <1>       cmp    ecx, [u.count] ; overrun check (!)
15624 000114BF 77DD              <1>       ja     short sysexec_14
15625 000114C1 8915[88030300]    <1>       mov    [u.count], edx
15626                            <1> sysexec_16:
15627 000114C7 A1[51040300]      <1>       mov    eax, [ii] ; first cluster
15628 000114CC E889000000        <1>       call   readi
15629 000114D1 8B0D[8C030300]    <1>       mov    ecx, [u.nread]
15630 000114D7 010D[90030300]    <1>       add    [u.break], ecx
15631                            <1> sysexec_17:
15632 000114DD A1[51040300]      <1>       mov    eax, [ii] ; first cluster
15633 000114E2 E886150000        <1>       call   iclose
15634 000114E7 31C0              <1>       xor     eax, eax
15635 000114E9 FEC0              <1>       inc    al
15636 000114EB 66A3[AA030300]    <1>       mov    [u.intr], ax ; 1 (interrupt/time-out is enabled)
15637 000114F1 66A3[AC030300]    <1>       mov    [u.quit], ax ; 1 ('crtl+brk' signal is enabled)
15638 000114F7 833D[BC030300]00  <1>         cmp  dword [u.ppgdir], 0  ; is the caller MainProg (kernel) ?
15639 000114FE 770C              <1>       ja     short sysexec_18 ; no, the caller is user process
15640                            <1>       ; If the caller is kernel (MainProg), 'sysexec' will come here
15641 00011500 8B15[F0880100]    <1>       mov    edx, [k_page_dir] ; kernel's page directory
15642 00011506 8915[BC030300]    <1>       mov    [u.ppgdir], edx ; next time 'sysexec' must not come here
15643                            <1> sysexec_18:
15644                            <1>       ; 02/05/2016
15645                            <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
15646                            <1>       ; 18/10/2015 (Retro UNIX 386 v1)
15647                            <1>       ; 05/08/2015
15648                            <1>       ; 29/07/2015
15649                            <1>
15650                            <1> ;     ; **** arguments list test start - 19/11/2017
15651                            <1> ;     mov    ebp, [argv]
15652                            <1> ;     sub    ebp, ECORE - 4096
15653                            <1> ;     add    ebp, [ecore]
15654                            <1> ;
15655                            <1> ;     mov    ebx, [ebp]
15656                            <1> ;     mov    [argc], bx
15657                            <1> ;     add    ebp, 4
15658                            <1> ;     mov    byte [ccolor], 1Fh
15659                            <1> ;_zx0:
15660                            <1> ;     cmp    word [argc], 0
15661                            <1> ;     jna    short _zx2
15662                            <1> ;_zx1:
15663                            <1> ;     push   ebp
15664                            <1> ;     mov    esi, [ebp]
15665                            <1> ;
15666                            <1> ;     sub    esi, ECORE - 4096
15667                            <1> ;     add    esi, [ecore]
15668                            <1> ;
15669                            <1> ;     call   print_cmsg
15670                            <1> ;
15671                            <1> ;     dec    word [argc]
15672                            <1> ;     jz     short _zx2
15673                            <1> ;
15674                            <1> ;     mov    al, '.'
15675                            <1> ;     mov    bl, 07h
15676                            <1> ;     mov    bh, [u.ttyn]
15677                            <1> ;     call   _write_tty
15678                            <1> ;
15679                            <1> ;     pop    ebp
15680                            <1> ;     add    ebp, 4
15681                            <1> ;     jmp    short _zx1
15682                            <1> ;_zx2:
15683                            <1> ;     pop    ebp
15684                            <1> ;     mov    byte [ccolor], 07h
15685                            <1> ;     mov    eax, 1
15686                            <1> ;     ; **** arguments list test stop
15687                            <1> ;     Test result is OK! (there is not a wrong thing) - 19/11/2017
15688                            <1>
15689 0001150C 8B2D[4C040300]    <1>       mov    ebp, [argv] ; user's stack pointer must point to argument
15690                            <1>                          ; list pointers (argument count)
15691 00011512 FA                <1>       cli
15692 00011513 8B25[8C880100]    <1>         mov     esp, [tss.esp0] ; ring 0 (kernel) stack pointer
15693                            <1>       ;mov         esp, [u.sp] ; Restore Kernel stack
15694                            <1>                             ; for this process
15695                            <1>       ;add   esp, 20 ; --> EIP, CS, EFLAGS, ESP, SS
15696                            <1>       ;xor   eax, eax ; 0
15697 00011519 FEC8              <1>       dec    al ; eax = 0
15698                            <1>       ;mov   edx, UDATA
15699                            <1>       ; 18/11/2017
15700 0001151B 6A23              <1>       push   UDATA ; user's stack segment
15701                            <1>       ;push  edx
15702 0001151D 55                <1>       push   ebp ; user's stack pointer
15703                            <1>                  ; (points to number of arguments)
15704                            <1>
15705                            <1>       ; 04/01/2017
15706                            <1>       ; MainProg comes here while [sysflg]= 0FFh
15707                            <1>       ; (but sysexec comes here while [sysflg]= 0)
15708 0001151E C605[5B030300]00  <1>       mov    byte [sysflg], 0 ; 04/01/2017
15709                            <1>                             ; (timer_int sysflg control)
15710 00011525 FB                <1>       sti
15711 00011526 9C                <1>       pushfd ; EFLAGS
15712                            <1>              ; Set IF for enabling interrupts in user mode
15713                            <1>       ;or    dword [esp], 200h
15714                            <1>       ;
15715                            <1>       ;mov   bx, UCODE
15716                            <1>       ;push  bx ; user's code segment
15717 00011527 6A1B              <1>       push   UCODE
15718                            <1>       ;push  0
15719 00011529 50                <1>       push   eax ; EIP (=0) - start address -
15720 0001152A 8925[5C030300]    <1>       mov    [u.sp], esp ; 29/07/2015
15721                            <1>       ; 05/08/2015
15722                            <1>       ; Remedy of a General Protection Fault during 'iretd' is here !
15723                            <1>       ; ('push dx' would cause to general protection fault,
15724                            <1>       ; after 'pop ds' etc.)
15725                            <1>       ;
```

```
15726                                  <1>       ;; push dx ; ds (UDATA)
15727                                  <1>       ;; push dx ; es (UDATA)
15728                                  <1>       ;; push dx ; fs (UDATA)
15729                                  <1>       ;; push dx ; gs (UDATA)
15730                                  <1>       ;
15731                                  <1>       ; This is a trick to prevent general protection fault
15732                                  <1>       ; during 'iretd' intruction at the end of 'sysrele' (in u1.s):
15733 00011530 66BA2300               <1>       mov    dx, UDATA ; 19/11/2017
15734 00011534 8EC2                   <1>       mov    es, dx ; UDATA
15735 00011536 06                     <1>       push   es ; ds (UDATA)
15736 00011537 06                     <1>       push   es ; es (UDATA)
15737 00011538 06                     <1>       push   es ; fs (UDATA)
15738 00011539 06                     <1>       push   es ; gs (UDATA)
15739 0001153A 66BA1000               <1>       mov    dx, KDATA
15740 0001153E 8EC2                   <1>       mov    es, dx
15741                                  <1>       ;
15742                                  <1>       ;; pushad simulation
15743 00011540 89E5                   <1>       mov    ebp, esp ; esp before pushad
15744 00011542 50                     <1>       push   eax ; eax (0)
15745 00011543 50                     <1>       push   eax ; ecx (0)
15746 00011544 50                     <1>       push   eax ; edx (0)
15747 00011545 50                     <1>       push   eax ; ebx (0)
15748 00011546 55                     <1>       push   ebp ; esp before pushad
15749 00011547 50                     <1>       push   eax ; ebp (0)
15750 00011548 50                     <1>       push   eax ; esi (0)
15751 00011549 50                     <1>       push   eax ; edi (0)
15752                                  <1>       ;
15753 0001154A A3[64030300]           <1>       mov    [u.r0], eax ; eax = 0
15754 0001154F 8925[60030300]         <1>       mov    [u.usp], esp
15755                                  <1>
15756                                  <1>       ; 14/11/2017
15757 00011555 E974C2FFFF             <1>       jmp    sysret0
15758                                  <1>
15759                                  <1> ;     ; 02/05/2016
15760                                  <1> ;     ;inc  byte [sysflg] ; 0FFh -> 0
15761                                  <1> ;     ;mov  byte [sysflg], 0 ; 04/01/2017
15762                                  <1> ;     movzx  ebx, byte [u.uno]
15763                                  <1> ;     shl    bl, 1 ; 13/11/2017
15764                                  <1> ;     cmp    word [ebx+p.ppid-2], 1 ; MainProg
15765                                  <1> ;     ja     sysret0 ; 03/05/2016
15766                                  <1> ;     push   sysret ; *
15767                                  <1> ;     mov    [u.usp], esp
15768                                  <1> ;     call   wswap ; save child process 'u' structure and
15769                                  <1> ;                  ; registers
15770                                  <1> ;     add    dword [u.usp], 4 ; 03/05/2016
15771                                  <1> ;sysexec_19: ; 02/05/2016
15772                                  <1> ;     retn ; * 'sysret' ; byte [sysflg] -> 0FFh
15773                                  <1>
15774                                  <1> readi:
15775                                  <1>       ; 01/05/2016
15776                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
15777                                  <1>       ; 20/05/2015 - Retro UNIX 386 v1
15778                                  <1>       ; 11/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
15779                                  <1>       ;
15780                                  <1>       ; Reads from a file whose the first cluster number in EAX
15781                                  <1>       ;
15782                                  <1>       ; INPUTS ->
15783                                  <1>       ;    EAX - First cluster number of the file
15784                                  <1>       ;    u.count - byte count user desires
15785                                  <1>       ;    u.base - points to user buffer
15786                                  <1>       ;    u.fofp - points to dword with current file offset
15787                                  <1>       ;    i.size - file size
15788                                  <1>       ;    cdev - logical dos drive number of the file
15789                                  <1>       ; OUTPUTS ->
15790                                  <1>       ;    u.count - cleared
15791                                  <1>       ;    u.nread - accumulates total bytes passed back
15792                                  <1>       ;
15793                                  <1>       ; ((EAX)) input/output
15794                                  <1>       ; (Retro UNIX Prototype : 14/12/2012 - 01/03/2013, UNIXCOPY.ASM)
15795                                  <1>         ; ((Modified registers: edx, ebx, ecx, esi, edi))
15796                                  <1>
15797 0001155A 31D2                   <1>       xor    edx, edx ; 0
15798 0001155C 8915[8C030300]         <1>       mov    [u.nread], edx ; 0
15799 00011562 668915[C4030300]       <1>       mov    [u.pcount], dx ; 19/05/2015
15800 00011569 3915[88030300]         <1>       cmp    [u.count], edx ; 0
15801 0001156F 7701                   <1>       ja     short readi_1
15802 00011571 C3                     <1>       retn
15803                                  <1> readi_1:
15804                                  <1> dskr:
15805                                  <1>       ; 01/05/2016
15806                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
15807                                  <1>       ; 24/05/2015 - 12/10/2015 (Retro UNIX 386 v1)
15808                                  <1>       ; 26/04/2013 - 03/08/2013 (Retro UNIX 8086 v1)
15809                                  <1> dskr_0:
15810 00011572 8B15[55040300]         <1>         mov  edx, [i.size]
15811 00011578 8B1D[74030300]         <1>       mov    ebx, [u.fofp]
15812 0001157E 2B13                   <1>       sub    edx, [ebx]
15813 00011580 7647                   <1>       jna    short dskr_4
15814                                  <1>       ;
15815 00011582 50                     <1>       push   eax ; 01/05/2016
15816 00011583 3B15[88030300]         <1>       cmp     edx, [u.count]
15817 00011589 7306                   <1>       jnb    short dskr_1
15818 0001158B 8915[88030300]         <1>       mov    [u.count], edx
15819                                  <1> dskr_1:
15820                                  <1>       ; EAX = First Cluster
15821                                  <1>       ; [Current_Drv] = Physical drive number
15822 00011591 E83B000000             <1>       call   mget_r
15823                                  <1>       ; NOTE: in 'mget_r', relevant sector will be read in buffer
15824                                  <1>       ; if it is not already in buffer !
15825 00011596 BB[8C050300]           <1>       mov    ebx, readi_buffer
15826 0001159B 803D[C6030300]00       <1>       cmp    byte [u.kcall], 0 ; the caller is 'namei' sign (=1)
15827 000115A2 770F                   <1>       ja     short dskr_3   ; zf=0 -> the caller is 'namei'
15828 000115A4 66833D[C4030300]00     <1>       cmp    word [u.pcount], 0
15829 000115AC 7705                   <1>       ja     short dskr_3
15830                                  <1> dskr_2:
```

```
15831                                  <1>       ; [u.base] = virtual address to transfer (as destination address)
15832 000115AE E894010000              <1>       call   trans_addr_w ; translate virtual address to physical (w)
15833                                  <1> dskr_3:
15834                                  <1>       ; EBX (r5) = system (I/O) buffer address -physical-
15835 000115B3 E8F7010000              <1>       call   sioreg
15836 000115B8 87F7                    <1>       xchg   esi, edi
15837                                  <1>       ; EDI = file (user data) offset
15838                                  <1>       ; ESI = sector (I/O) buffer offset
15839                                  <1>       ; ECX = byte count
15840 000115BA F3A4                    <1>       rep    movsb
15841                                  <1>       ; eax = remain bytes in buffer
15842                                  <1>         ;       (check if remain bytes in the buffer > [u.pcount])
15843 000115BC 09C0                    <1>       or     eax, eax
15844 000115BE 75EE                    <1>       jnz    short dskr_2 ; (page end before system buffer end!)
15845 000115C0 58                      <1>       pop    eax  ; (first cluster number)
15846 000115C1 390D[88030300]          <1>       cmp    [u.count], ecx ; 0
15847 000115C7 77A9                    <1>       ja     short dskr_0
15848                                  <1> dskr_4:
15849 000115C9 C605[C6030300]00        <1>       mov    byte [u.kcall], 0
15850 000115D0 C3                      <1>       retn
15851                                  <1>
15852                                  <1> mget_r:
15853                                  <1>       ; 24/10/2016
15854                                  <1>       ; 22/10/2016
15855                                  <1>       ; 12/10/2016
15856                                  <1>       ; 29/04/2016
15857                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
15858                                  <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
15859                                  <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
15860                                  <1>       ;
15861                                  <1>       ; Get existing or (allocate) a new disk block for file
15862                                  <1>       ;
15863                                  <1>       ; INPUTS ->
15864                                  <1>       ;    [u.fofp] = file offset pointer
15865                                  <1>       ;    EAX = First Cluster
15866                                  <1>       ;    [cdev] = Logical dos drive number
15867                                  <1>       ;    ([u.off] = file offset)
15868                                  <1>       ; OUTPUTS ->
15869                                  <1>       ;    EAX = logical sector number
15870                                  <1>       ;    ESI = Logical Dos Drive Description Table address
15871                                  <1>       ;
15872                                  <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI
15873                                  <1>
15874 000115D1 8B35[74030300]          <1>       mov     esi, [u.fofp]
15875 000115D7 8B1E                    <1>       mov    ebx, [esi] ; (u.off)
15876                                  <1>
15877 000115D9 29C9                    <1>       sub    ecx, ecx
15878 000115DB 8A2D[46030300]          <1>       mov    ch, [cdev]
15879                                  <1>
15880 000115E1 BE00010900              <1>       mov    esi, Logical_DOSDisks
15881 000115E6 01CE                    <1>       add    esi, ecx
15882                                  <1>
15883 000115E8 380D[24950100]          <1>       cmp    [readi.valid], cl ; 0
15884 000115EE 7649                    <1>       jna    short mget_r_0
15885                                  <1>
15886 000115F0 3A2D[25950100]          <1>       cmp    ch, [readi.drv]
15887 000115F6 7541                    <1>       jne    short mget_r_0
15888                                  <1>
15889 000115F8 3B05[38950100]          <1>       cmp    eax, [readi.fclust]
15890 000115FE 7565                    <1>       jne    short mget_r_3
15891                                  <1>
15892 00011600 89D8                    <1>       mov    eax, ebx ; file offset
15893 00011602 668B0D[2C950100]        <1>       mov    cx, [readi.bpc]
15894 00011609 41                      <1>       inc    ecx ; <= 65536
15895 0001160A 29D2                    <1>       sub    edx, edx
15896 0001160C F7F1                    <1>       div    ecx
15897                                  <1>
15898 0001160E 8B3D[34950100]          <1>       mov    edi, [readi.c_index] ; cluster index
15899                                  <1>
15900 00011614 39F8                    <1>       cmp    eax, edi
15901 00011616 757A                    <1>         jne     short mget_r_4  ; (*)
15902                                  <1>
15903                                  <1>       ; edx = byte offset in cluster (<= 65535)
15904 00011618 668915[2E950100]        <1>       mov    [readi.offset], dx
15905 0001161F 66C1EA09                <1>       shr    dx, 9 ; / 512
15906 00011623 8815[27950100]          <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
15907                                  <1>
15908 00011629 A1[30950100]            <1>       mov    eax, [readi.cluster]  ; > 0 if [readi.valid] = 1
15909 0001162E 8B15[3C950100]          <1>       mov    edx, [readi.fs_index]
15910 00011634 E99A000000              <1>         jmp     mget_r_7
15911                                  <1>
15912                                  <1> mget_r_0:
15913 00011639 882D[25950100]          <1>       mov    [readi.drv], ch ; physical drive number
15914 0001163F 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
15915 00011643 7707                    <1>       ja     short mget_r_1
15916 00011645 8A4E12                  <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
15917 00011648 D0E9                    <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
15918 0001164A EB03                    <1>       jmp    short mget_r_2
15919                                  <1> mget_r_1:
15920 0001164C 8A4E13                  <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
15921                                  <1> mget_r_2:
15922 0001164F 880D[26950100]          <1>       mov    [readi.spc], cl  ; sectors per cluster
15923                                  <1>       ; NOTE: readi bytes per sector value is always 512 !
15924 00011655 66C1E109                <1>       shl    cx, 9 ; * 512
15925 00011659 6649                    <1>       dec    cx ; bytes per cluster - 1
15926 0001165B 66890D[2C950100]        <1>       mov    [readi.bpc], cx
15927 00011662 6629C9                  <1>       sub    cx, cx
15928                                  <1> mget_r_3:
15929 00011665 A3[38950100]            <1>       mov    [readi.fclust], eax ; first cluster (or FDT address)
15930 0001166A 880D[24950100]          <1>       mov    [readi.valid], cl ; 0
15931                                  <1>       ;mov   [readi.s_index], cl ; 0
15932                                  <1>       ;mov   [readi.offset], cx ; 0
15933 00011670 890D[34950100]          <1>       mov    [readi.c_index], ecx ; 0
15934 00011676 890D[30950100]          <1>       mov    [readi.cluster], ecx ; 0
15935 0001167C 890D[28950100]          <1>       mov    [readi.sector], ecx ; 0
```

```
15936                                  <1>
15937 00011682 89D8                    <1>       mov    eax, ebx ; file offset
15938 00011684 668B0D[2C950100]        <1>       mov    cx, [readi.bpc]
15939 0001168B 41                      <1>       inc    ecx ; <= 65536
15940 0001168C 29D2                    <1>       sub    edx, edx
15941 0001168E F7F1                    <1>       div    ecx
15942                                  <1>       ;mov   edi, [readi.c_index] ; previous cluster index
15943 00011690 29FF                    <1>       sub    edi, edi
15944                                  <1> mget_r_4:
15945 00011692 A3[34950100]            <1>       mov    [readi.c_index], eax ; cluster index
15946                                  <1>       ; edx = byte offset in cluster (<= 65535)
15947 00011697 668915[2E950100]        <1>       mov    [readi.offset], dx
15948 0001169E 66C1EA09                <1>       shr    dx, 9 ; / 512
15949 000116A2 8815[27950100]          <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
15950                                  <1>
15951 000116A8 89C1                    <1>       mov    ecx, eax ; current cluster index
15952 000116AA A1[38950100]            <1>       mov    eax, [readi.fclust]
15953 000116AF 09C9                    <1>       or     ecx, ecx ; cluster index
15954 000116B1 741B                    <1>       jz     short mget_r_6
15955                                  <1>
15956 000116B3 39CF                    <1>       cmp    edi, ecx
15957 000116B5 7710                    <1>       ja     short mget_r_5 ; old cluster index is higher
15958 000116B7 8B15[30950100]          <1>       mov    edx, [readi.cluster]
15959 000116BD 21D2                    <1>       and    edx, edx
15960 000116BF 7406                    <1>       jz     short mget_r_5
15961                                  <1>       ; valid 'readi' parameters (*)
15962 000116C1 89D0                    <1>       mov    eax, edx
15963 000116C3 29F9                    <1>       sub    ecx, edi
15964 000116C5 740C                    <1>       jz     short mget_r_7
15965                                  <1> mget_r_5:
15966                                  <1>       ; EAX = Beginning cluster
15967                                  <1>       ; EDX = Sector index in disk/file section
15968                                  <1>       ;      (Only for SINGLIX file system!)
15969                                  <1>       ; ECX = Cluster sequence number after the beginning cluster
15970                                  <1>       ; ESI = Logical DOS Drive Description Table address
15971 000116C7 E879BFFFFF              <1>       call   get_cluster_by_index
15972 000116CC 724E                    <1>       jc     short mget_r_err
15973                                  <1>       ; EAX = Cluster number
15974                                  <1> mget_r_6:
15975 000116CE A3[30950100]            <1>       mov    [readi.cluster], eax ; FDT number for Singlix File System
15976                                  <1> mget_r_7:
15977 000116D3 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
15978 000116D7 765F                    <1>       jna    short mget_r_12
15979                                  <1>
15980 000116D9 83E802                  <1>       sub    eax, 2
15981 000116DC 0FB615[26950100]        <1>       movzx  edx, byte [readi.spc]
15982 000116E3 F7E2                    <1>       mul    edx
15983                                  <1>
15984 000116E5 034668                  <1>       add    eax, [esi+LD_DATABegin]
15985 000116E8 8A15[27950100]          <1>       mov    dl, [readi.s_index]
15986 000116EE 01D0                    <1>       add    eax, edx
15987                                  <1> mget_r_8:
15988                                  <1>       ; eax = logical sector number
15989 000116F0 803D[24950100]00        <1>       cmp    byte [readi.valid], 0
15990 000116F7 7608                    <1>       jna    short mget_r_9
15991 000116F9 3B05[28950100]          <1>       cmp    eax, [readi.sector]
15992 000116FF 7436                    <1>       je     short mget_r_11 ; sector is already in 'readi' buffer
15993                                  <1> mget_r_9:
15994 00011701 A3[28950100]            <1>       mov    [readi.sector], eax
15995 00011706 BB[8C050300]            <1>       mov    ebx, readi_buffer ; buffer address
15996 0001170B B901000000              <1>       mov    ecx, 1
15997                                  <1>       ; 29/04/2016
15998                                  <1>       ;xor   dl, dl
15999                                  <1>
16000                                  <1>       ; EAX = Logical sector number
16001                                  <1>       ; ECX = Sector count
16002                                  <1>       ; EBX = Buffer address
16003                                  <1>       ; (EDX = 0)
16004                                  <1>       ; ESI = Logical DOS drive description table address
16005                                  <1>
16006 00011710 E868130000              <1>       call   disk_read
16007 00011715 7314                    <1>       jnc    short mget_r_10
16008                                  <1>
16009                                  <1>       ; 22/10/2016 (15h -> 17)
16010 00011717 B811000000              <1>       mov    eax, 17 ; Drive not ready or read error !
16011                                  <1> mget_r_err:
16012 0001171C A3[C8030300]            <1>       mov    [u.error], eax
16013                                  <1>       ; 12/10/2016
16014 00011721 A3[64030300]            <1>       mov    [u.r0], eax
16015 00011726 E981C0FFFF              <1>       jmp    error
16016                                  <1> mget_r_10:
16017 0001172B C605[24950100]01        <1>       mov    byte [readi.valid], 1 ; 24/10/2016
16018 00011732 A1[28950100]            <1>       mov    eax, [readi.sector]
16019                                  <1> mget_r_11:
16020 00011737 C3                      <1>       retn
16021                                  <1> mget_r_12:
16022                                  <1>       ; EAX = FDT number
16023                                  <1>       ; EDX = Sector index from FDT sector (0,1,2,3,4...)
16024 00011738 40                      <1>       inc    eax ; the first data sector in FS disk section
16025 00011739 8915[3C950100]          <1>       mov    [readi.fs_index], edx
16026 0001173F 01D0                    <1>       add    eax, edx
16027 00011741 EBAD                    <1>       jmp    short mget_r_8
16028                                  <1>
16029                                  <1> trans_addr_r:
16030                                  <1>       ; 12/10/2016
16031                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
16032                                  <1>       ; Translate virtual address to physical address
16033                                  <1>       ; for reading from user's memory space
16034                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
16035                                  <1>
16036 00011743 31D2                    <1>       xor    edx, edx ; 0 (read access sign)
16037 00011745 EB04                    <1>       jmp    short trans_addr_rw
16038                                  <1>
16039                                  <1> trans_addr_w:
16040                                  <1>       ; 12/10/2016
```

```
16041                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16042                                  <1>       ; Translate virtual address to physical address
16043                                  <1>       ; for writing to user's memory space
16044                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
16045                                  <1>
16046 00011747 29D2                   <1>       sub    edx, edx
16047 00011749 FEC2                   <1>       inc    dl ; 1 (write access sign)
16048                                  <1> trans_addr_rw:
16049 0001174B 50                     <1>       push   eax
16050 0001174C 53                     <1>       push   ebx
16051 0001174D 52                     <1>       push   edx ; r/w sign (in DL)
16052                                  <1>       ;
16053 0001174E 8B1D[84030300]         <1>       mov    ebx, [u.base]
16054 00011754 E8634AFFFF             <1>       call   get_physical_addr ; get physical address
16055 00011759 730F                   <1>       jnc    short passc_0
16056 0001175B A3[C8030300]           <1>       mov    [u.error], eax
16057 00011760 A3[64030300]           <1>       mov    [u.r0], eax ; 12/10/2016
16058                                  <1>       ;pop   edx
16059                                  <1>       ;pop   ebx
16060                                  <1>       ;pop   eax
16061 00011765 E942C0FFFF             <1>       jmp    error
16062                                  <1> passc_0:
16063 0001176A F6C202                 <1>       test   dl, PTE_A_WRITE ; writable page
16064 0001176D 5A                     <1>       pop    edx
16065 0001176E 751C                   <1>       jnz    short passc_1
16066                                  <1>
16067 00011770 20D2                   <1>       and    dl, dl
16068 00011772 7418                   <1>       jz     short passc_1
16069                                  <1>       ; read only (duplicated) page -must be copied to a new page-
16070                                  <1>       ; EBX = linear address
16071 00011774 51                     <1>       push   ecx
16072 00011775 E8DB46FFFF             <1>       call   copy_page
16073 0001177A 59                     <1>       pop    ecx
16074 0001177B 721E                   <1>       jc     short passc_2
16075 0001177D 50                     <1>       push   eax ; physical address of the new/allocated page
16076 0001177E E86349FFFF             <1>       call   add_to_swap_queue
16077 00011783 58                     <1>       pop    eax
16078 00011784 81E3FF0F0000           <1>       and    ebx, PAGE_OFF ; 0FFFh
16079                                  <1>       ;mov   ecx, PAGE_SIZE
16080                                  <1>       ;sub   ecx, ebx
16081 0001178A 01D8                   <1>       add    eax, ebx
16082                                  <1> passc_1:
16083 0001178C A3[C0030300]           <1>       mov    [u.pbase], eax ; physical address
16084 00011791 66890D[C4030300]       <1>       mov    [u.pcount], cx ; remain byte count in page (1-4096)
16085 00011798 5B                     <1>       pop    ebx
16086 00011799 58                     <1>       pop    eax
16087 0001179A C3                     <1>       retn
16088                                  <1> passc_2:
16089 0001179B B804000000             <1>       mov    eax, ERR_MINOR_IM ; "Insufficient memory !" error
16090 000117A0 A3[64030300]           <1>       mov    [u.r0], eax ; 12/10/2016
16091 000117A5 A3[C8030300]           <1>       mov    dword [u.error], eax
16092                                  <1>       ;pop   ebx
16093                                  <1>       ;pop   eax
16094 000117AA E9FDBFFFFF             <1>       jmp    error
16095                                  <1>
16096                                  <1> sioreg:
16097                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16098                                  <1>       ; 19/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
16099                                  <1>       ; 12/03/2013 - 22/07/2013 (Retro UNIX 8086 v1)
16100                                  <1>       ; INPUTS ->
16101                                  <1>       ;     EBX = system buffer (data) address (r5)
16102                                  <1>       ;     [u.fofp] = pointer to file offset pointer
16103                                  <1>       ;     [u.base] = virtual address of the user buffer
16104                                  <1>       ;     [u.pbase] = physical address of the user buffer
16105                                  <1>       ;     [u.count] = byte count
16106                                  <1>       ;     [u.pcount] = byte count within page frame
16107                                  <1>       ; OUTPUTS ->
16108                                  <1>       ;     ESI = user data offset (r1)
16109                                  <1>       ;     EDI = system (I/O) buffer offset (r2)
16110                                  <1>       ;     ECX = byte count (r3)
16111                                  <1>       ;     EAX = remain bytes after byte count within page frame
16112                                  <1>       ;      (If EAX > 0, transfer will continue from the next page)
16113                                  <1>         ;
16114                                  <1>       ; ((Modified registers:  EDX))
16115                                  <1>
16116 000117AF 8B35[74030300]         <1>         mov     esi, [u.fofp]
16117 000117B5 8B3E                   <1>         mov     edi, [esi]
16118 000117B7 89F9                   <1>       mov    ecx, edi
16119 000117B9 81C900FEFFFF           <1>       or     ecx, 0FFFFFE00h
16120 000117BF 81E7FF010000           <1>       and    edi, 1FFh
16121 000117C5 01DF                   <1>       add    edi, ebx ; EBX = system buffer (data) address
16122 000117C7 F7D9                   <1>       neg    ecx
16123 000117C9 3B0D[88030300]         <1>       cmp    ecx, [u.count]
16124 000117CF 7606                   <1>       jna    short sioreg_0
16125 000117D1 8B0D[88030300]         <1>       mov    ecx, [u.count]
16126                                  <1> sioreg_0:
16127 000117D7 803D[C6030300]00       <1>       cmp    byte [u.kcall], 0
16128 000117DE 7613                   <1>       jna    short sioreg_1
16129                                  <1>        ; the caller is 'mkdir' or 'namei'
16130 000117E0 A1[84030300]           <1>       mov    eax, [u.base]
16131 000117E5 A3[C0030300]           <1>       mov    [u.pbase], eax ; physical address = virtual address
16132 000117EA 66890D[C4030300]       <1>       mov    word [u.pcount], cx ; remain bytes in buffer (1 sector)
16133 000117F1 EB0B                   <1>       jmp    short sioreg_2
16134                                  <1> sioreg_1:
16135 000117F3 0FB715[C4030300]       <1>       movzx  edx, word [u.pcount]
16136 000117FA 39D1                   <1>       cmp    ecx, edx
16137 000117FC 772A                   <1>       ja     short sioreg_4 ; transfer count > [u.pcount]
16138                                  <1> sioreg_2: ; 2:
16139 000117FE 31C0                   <1>       xor    eax, eax
16140                                  <1> sioreg_3:
16141 00011800 010D[8C030300]         <1>       add    [u.nread], ecx
16142 00011806 290D[88030300]         <1>       sub    [u.count], ecx
16143 0001180C 010D[84030300]         <1>       add    [u.base], ecx
16144 00011812 010E                   <1>         add  [esi], ecx
16145 00011814 8B35[C0030300]         <1>       mov    esi, [u.pbase]
```

```
16146 0001181A 66290D[C4030300]    <1>       sub    [u.pcount], cx
16147 00011821 010D[C0030300]      <1>       add    [u.pbase], ecx
16148 00011827 C3                  <1>         retn
16149                              <1> sioreg_4:
16150                              <1>       ; transfer count > [u.pcount]
16151                              <1>       ; (ecx > edx)
16152 00011828 89C8                <1>       mov    eax, ecx
16153 0001182A 29D0                <1>       sub    eax, edx ; remain bytes for 1 sector (block) transfer
16154 0001182C 89D1                <1>       mov    ecx, edx ; current transfer count = [u.pcount]
16155 0001182E EBD0                <1>       jmp    short sioreg_3
16156                              <1>
16157                              <1> tswitch: ; Retro UNIX 386 v1
16158                              <1> tswap:
16159                              <1>       ; 16/01/2017
16160                              <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
16161                              <1>       ; 10/05/2015 - 01/09/2015 (Retro UNIX 386 v1)
16162                              <1>       ; 14/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
16163                              <1>       ; time out swap, called when a user times out.
16164                              <1>       ; the user is put on the low priority queue.
16165                              <1>       ; This is done by making a link from the last user
16166                              <1>       ; on the low priority queue to him via a call to 'putlu'.
16167                              <1>       ; then he is swapped out.
16168                              <1>
16169                              <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 21/05/2016 **
16170                              <1>       ;     * when a high priority (event) process will be stopped
16171                              <1>       ;      (swapped out, swithched out/off), 'tswap/tswitch' will
16172                              <1>       ;      not add it to a run queue.
16173                              <1>       ;      /// What for: Process may be already in a run queue,
16174                              <1>       ;      it is unspeficied state because process might be started
16175                              <1>       ;      by a timer event which does not regard previous priority
16176                              <1>       ;      level and run queue of the process (for fast executing!).
16177                              <1>       ;      After the 'run for event', process will be sequenced
16178                              <1>       ;      to run by it's actual run queue. ///
16179                              <1>       ;
16180                              <1>       ; Retro UNIX 386 v1 modification ->
16181                              <1>       ;       swap (software task switch) is performed by changing
16182                              <1>       ;      user's page directory (u.pgdir) instead of segment change
16183                              <1>       ;      as in Retro UNIX 8086 v1.
16184                              <1>       ;
16185                              <1>       ; RETRO UNIX 8086 v1 modification ->
16186                              <1>       ;       'swap to disk' is replaced with 'change running segment'
16187                              <1>       ;      according to 8086 cpu (x86 real mode) architecture.
16188                              <1>       ;      pdp-11 was using 64KB uniform memory while IBM PC
16189                              <1>       ;      compatibles was using 1MB segmented memory
16190                              <1>       ;      in 8086/8088 times.
16191                              <1>       ;
16192                              <1>       ; INPUTS ->
16193                              <1>       ;    u.uno - users process number
16194                              <1>       ;    runq+4 - lowest priority queue
16195                              <1>       ; OUTPUTS ->
16196                              <1>       ;    r0 - users process number
16197                              <1>       ;    r2 - lowest priority queue address
16198                              <1>       ;
16199                              <1>       ; ((AX = R0, BX = R2)) output
16200                              <1>       ; ((Modified registers: EDX, EBX, ECX, ESI, EDI))
16201                              <1>       ;
16202                              <1>
16203                              <1>       NOTE:
16204                              <1>       ;* [u.pri] priority level is specified by run queue which is process
16205                              <1>       ;  comes to run from.
16206                              <1>       ;* Initial [u.pri] is 1 ('normal/regular') for programs
16207                              <1>       ;  (which are launched by MainProg or 'sysexec'), it is changed
16208                              <1>       ;  to 2 ('high') by timer event, if program uses 'systimer' system call.
16209                              <1>       ;* Program (Process) also can change it's running priority
16210                              <1>       ;  from 1 to 0 or up to 2 by using 'syspri' system call; but,
16211                              <1>       ;  if program selects priority level 2 (high) for running, next time
16212                              <1>       ;  it is reduced to 1 (normal/regular) because 'syspri' adds this
16213                              <1>       ;  program to 'run for normal' queue while running duration is a bit
16214                              <1>       ;  protected from swap/switch out immediate, behalf of other high
16215                              <1>       ;  priority process in sequence. Program (with high priority) will not
16216                              <1>       ;  be swapped/switched out (by timer event) before it's time quantum
16217                              <1>       ;  will be elapsed, but, this will be temporary if program is not using
16218                              <1>       ;  timer event function.
16219                              <1>
16220                              <1>       ;For example:
16221                              <1>       ;If a process frequently gets a timer event, it runs at high priority
16222                              <1>       ;level but when it returns from running it returns to actual run queue,
16223                              <1>       ;not to 'run for event' queue again.
16224                              <1>       ;'tswap' will not change the sequence at return/stop(swap out) stage.
16225                              <1>       ;But if priority level not high (=2, 'run for event'), 'tswap/tswitch'
16226                              <1>       ;will add the stopping process to relevant run queue according to
16227                              <1>       ;[u.pri] priority level.
16228                              <1>
16229                              <1>       ; 16/01/2017
16230 00011830 BB[54030300]        <1>       mov    ebx, runq+2  ; 'runq_normal' ; normal/regular priority
16231                              <1>       ; 21/05/2016
16232                              <1>       ;cmp   byte [u.pri], 2     ; high priority (run for event) ?
16233                              <1>       ;jnb   short swap
16234                              <1>       ; 16/01/2017
16235                              <1>       ; (Normal and also high/event priority processes will be added to
16236                              <1>       ; normal priority run queue for ensuring circular running sequence!)
16237                              <1>       ; (Timer interrupt or 'syspri' system call may change priority and run
16238                              <1>       ; queue to high/event level.)
16239 00011835 803D[A9030300]00    <1>       cmp    byte [u.pri], 0
16240 0001183C 7702                <1>       ja     short tswap_1; normal priority run queue
16241                              <1>       ;
16242 0001183E 43                  <1>       inc    ebx
16243 0001183F 43                  <1>       inc    ebx          ; runq+4, 'runq_background', low priority
16244                              <1> tswap_1:
16245 00011840 A0[B3030300]        <1>       mov    al, [u.uno]
16246                              <1>                     ; movb u.uno,r1 / move users process number to r1
16247                              <1>             ; mov  $runq+4,r2
16248                              <1>                     ; / move lowest priority queue address to r2
16249                              <1>             ; ebx = run queue
16250 00011845 E8FE000000          <1>       call   putlu
```

```
16251                                 <1>             ; jsr r0,putlu / create link from last user on Q to
16252                                 <1>                             ; / u.uno's user
16253                                 <1>
16254                                 <1> switch: ; Retro UNIX 386 v1
16255                                 <1> swap:
16256                                 <1>       ; 02/01/2017
16257                                 <1>       ; 21/05/2016
16258                                 <1>       ; 20/05/2016
16259                                 <1>       ; 02/05/2016
16260                                 <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16261                                 <1>       ; 10/05/2015 - 02/09/2015 (Retro UNIX 386 v1)
16262                                 <1>       ; 14/04/2013 - 08/03/2014 (Retro UNIX 8086 v1)
16263                                 <1>       ;
16264                                 <1>       ; 'swap' is routine that controls the swapping of processes
16265                                 <1>       ; in and out of core.
16266                                 <1>       ;
16267                                 <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 20/05/2016 **
16268                                 <1>       ;     * 3 different priority level is applied
16269                                 <1>       ;      (just as original unix v1)
16270                                 <1>       ;      1) high priority (event) run queue, 'runq_event'
16271                                 <1>       ;      2) normal priority (regular) run queue, 'runq_normal'
16272                                 <1>       ;      3) low priority (background) run queue, 'runq_backgroud'
16273                                 <1>       ;      'swap' code will run a process which has max. priority
16274                                 <1>         ;       (for earliest event at first)
16275                                 <1>       ;
16276                                 <1>       ; Retro UNIX 386 v1 modification ->
16277                                 <1>       ;       swap (software task switch) is performed by changing
16278                                 <1>       ;     user's page directory (u.pgdir) instead of segment change
16279                                 <1>       ;     as in Retro UNIX 8086 v1.
16280                                 <1>       ;
16281                                 <1>       ; RETRO UNIX 8086 v1 modification ->
16282                                 <1>       ;       'swap to disk' is replaced with 'change running segment'
16283                                 <1>       ;     according to 8086 cpu (x86 real mode) architecture.
16284                                 <1>       ;      pdp-11 was using 64KB uniform memory while IBM PC
16285                                 <1>       ;      compatibles was using 1MB segmented memory
16286                                 <1>       ;      in 8086/8088 times.
16287                                 <1>       ;
16288                                 <1>       ; INPUTS ->
16289                                 <1>       ;    runq table - contains processes to run.
16290                                 <1>       ;    p.link - contains next process in line to be run.
16291                                 <1>       ;    u.uno - process number of process in core
16292                                 <1>       ;    s.stack - swap stack used as an internal stack for swapping.
16293                                 <1>       ; OUTPUTS ->
16294                                 <1>       ;    (original unix v1 -> present process to its disk block)
16295                                 <1>       ;    (original unix v1 -> new process into core ->
16296                                 <1>       ;        Retro Unix 8086 v1 -> segment registers changed
16297                                 <1>       ;        for new process)
16298                                 <1>       ;    u.quant = 3 (Time quantum for a process)
16299                                 <1>       ;     ((INT 1Ch count down speed -> 18.2 times per second)
16300                                 <1>       ;    RETRO UNIX 8086 v1 will use INT 1Ch (18.2 times per second)
16301                                 <1>       ;       for now, it will swap the process if there is not
16302                                 <1>       ;       a keyboard event (keystroke) (Int 15h, function 4Fh)
16303                                 <1>       ;       or will count down from 3 to 0 even if there is a
16304                                 <1>       ;         keyboard event locking due to repetitive key strokes.
16305                                 <1>       ;       u.quant will be reset to 3 for RETRO UNIX 8086 v1.
16306                                 <1>       ;
16307                                 <1>       ; ((Modified registers: EAX, EDX, EBX, ECX, ESI, EDI))
16308                                 <1>
16309                                 <1>       ;NOTE:
16310                                 <1>       ;High priority queue is the first for selecting a process to run.
16311                                 <1>       ;If there is not a process in high priority level run queue,
16312                                 <1>       ;a process in normal priority run queue will be selected
16313                                 <1>       ;or a proces in low priority run queue will be selected if normal
16314                                 <1>       ;priority level run queue is empty.
16315                                 <1>
16316                                 <1>       ; 21/05/2016 -(3 priority levels, 3 run queues)
16317 0001184A BE[52030300]          <1>       mov   esi, runq ; 'runq_event' ; high priority, 'run for event'
16318 0001184F C605[80950100]03      <1>       mov   byte [priority], 3 ; high priority + 1
16319 00011856 31DB                  <1>       xor   ebx, ebx ; 02/01/2017
16320                                <1> swap_0: ; 1: / search runq table for highest priority process
16321 00011858 66AD                  <1>       lodsw  ; mov ax, [esi], add esi+2
16322                                <1>       ;xor   ebx, ebx ; 02/05/2016
16323 0001185A 6621C0                <1>       and   ax, ax ; are there any processes to run in this Q entry
16324 0001185D 750E                  <1>       jnz   short swap_2
16325                                <1>       ; 21/05/2026
16326                                <1>       ; runq_normal = runq+2, runq_background = runq+4
16327 0001185F FE0D[80950100]        <1>       dec   byte [priority] ; 3 -> 3, 2 -> 1, 1-> 0
16328 00011865 75F1                  <1>       jnz   short swap_0
16329                                <1>       ;cmp   esi, runq+6  ; if zero compare address to end of table
16330                                <1>       ;jb    short swap_0 ; if not at end, go back
16331                                <1> swap_1:
16332                                <1>       ; 02/05/2016
16333                                <1>       ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
16334                                <1>       ; No user process to run...
16335                                <1>       ; Run the kernel process... MainProg: Internal Command Interpreter
16336 00011867 FEC0                  <1>       inc   al ; mov al, 1  ; process number of MainProg
16337 00011869 FEC3                  <1>       inc   bl ; mov bl, al ; 1
16338 0001186B EB1E                  <1>       jmp   short swap_4
16339                                <1> swap_2:
16340                                <1>       ; 21/05/2016
16341 0001186D FE0D[80950100]        <1>       dec   byte [priority] ; priority level of present user/process
16342                                <1>                             ; 0, 1, 2
16343 00011873 4E                    <1>       dec   esi
16344 00011874 4E                    <1>         dec     esi
16345                                <1>       ;
16346 00011875 88C3                  <1>       mov   bl, al
16347 00011877 38E0                  <1>       cmp   al, ah ; is there only 1 process in the queue to be run
16348 00011879 740A                  <1>       je    short swap_3 ; yes
16349 0001187B 8AA3[9F000300]        <1>       mov   ah, [ebx+p.link-1]
16350 00011881 8826                  <1>             mov   [esi], ah ; move next process in line into run queue
16351 00011883 EB06                  <1>       jmp   short swap_4
16352                                <1> swap_3:
16353 00011885 6631D2                <1>       xor   dx, dx
16354 00011888 668916                <1>       mov   [esi], dx ; zero the entry; no processes on the Q
16355                                <1> swap_4:
```

```
16356 0001188B 8A25[B3030300]      <1>       mov    ah, [u.uno]
16357 00011891 38C4                <1>       cmp    ah, al ; is this process the same as the process in core?
16358 00011893 743B                <1>              je     short swap_8 ; yes, don't have to swap
16359 00011895 08E4                <1>       or     ah, ah  ; is the process # = 0
16360 00011897 740D                <1>              jz     short swap_6 ; 'sysexit'
16361                              <1>       ;cmp   ah, al ;is this process the same as the process in core?
16362                              <1>              ;je    short swap_8 ; yes, don't have to swap
16363 00011899 8925[60030300]      <1>       mov    [u.usp], esp ; return  address for 'syswait' & 'sleep'
16364 0001189F E834000000          <1>       call   wswap   ; write out core to disk
16365 000118A4 EB1C                <1>       jmp    short swap_7
16366                              <1> swap_6:
16367                              <1>       ; Deallocate memory pages belong to the process
16368                              <1>       ; which is being terminated.
16369                              <1>       ; (Retro UNIX 386 v1 modification !)
16370                              <1>       ;
16371 000118A6 53                 <1>       push   ebx
16372 000118A7 A1[B8030300]        <1>       mov    eax, [u.pgdir]  ; page directory of the process
16373 000118AC 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
16374 000118B2 E82943FFFF          <1>       call   deallocate_page_dir
16375 000118B7 A1[B4030300]        <1>       mov    eax, [u.upage] ; 'user' structure page of the process
16376 000118BC E8C443FFFF          <1>       call   deallocate_page
16377 000118C1 5B                  <1>       pop    ebx
16378                              <1> swap_7:
16379 000118C2 C0E302              <1>       shl    bl, 2 ; * 4
16380 000118C5 8B83[BC000300]      <1>       mov    eax, [ebx+p.upage-4] ; the 'u' page of the new process
16381 000118CB E840000000          <1>       call   rswap ; read new process into core
16382                              <1> swap_8:
16383                              <1>       ; Retro UNIX  8086 v1 modification !
16384 000118D0 C605[A8030300]04    <1>       mov    byte [u.quant], time_count
16385 000118D7 C3                  <1>       retn
16386                              <1>
16387                              <1> wswap:  ; < swap out, swap to disk >
16388                              <1>       ; 28/02/2017 (fnsave)
16389                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16390                              <1>       ; 09/05/2015 (Retro UNIX 386 v1)
16391                              <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
16392                              <1>       ; 'wswap' writes out the process that is in core onto its
16393                              <1>       ; appropriate disk area.
16394                              <1>       ;
16395                              <1>       ; Retro UNIX 386 v1 modification ->
16396                              <1>       ;       User (u) structure content and the user's register content
16397                              <1>       ;     will be copied to the process's/user's UPAGE (a page for
16398                              <1>       ;     saving 'u' structure and user registers for task switching).
16399                              <1>       ;     u.usp - points to kernel stack address which contains
16400                              <1>       ;             user's registers while entering system call.
16401                              <1>       ;     u.sp  - points to kernel stack address
16402                              <1>       ;             to return from system call -for IRET-.
16403                              <1>       ;     [u.usp]+32+16 = [u.sp]
16404                              <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
16405                              <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
16406                              <1>       ;
16407                              <1>       ; Retro UNIX 8086 v1 modification ->
16408                              <1>       ;       'swap to disk' is replaced with 'change running segment'
16409                              <1>       ;     according to 8086 cpu (x86 real mode) architecture.
16410                              <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
16411                              <1>       ;     compatibles was using 1MB segmented memory
16412                              <1>       ;     in 8086/8088 times.
16413                              <1>       ;
16414                              <1>       ; INPUTS ->
16415                              <1>       ;    u.break - points to end of program
16416                              <1>       ;    u.usp - stack pointer at the moment of swap
16417                              <1>       ;    core - beginning of process program
16418                              <1>       ;    ecore - end of core
16419                              <1>       ;    user - start of user parameter area
16420                              <1>       ;    u.uno - user process number
16421                              <1>       ;    p.dska - holds block number of process
16422                              <1>       ; OUTPUTS ->
16423                              <1>       ;    swp I/O queue
16424                              <1>       ;    p.break - negative word count of process
16425                              <1>       ;    r1 - process disk address
16426                              <1>       ;    r2 - negative word count
16427                              <1>       ;
16428                              <1>       ; RETRO UNIX 8086 v1 input/output:
16429                              <1>       ;
16430                              <1>       ; INPUTS ->
16431                              <1>       ;    u.uno - process number (to be swapped out)
16432                              <1>       ; OUTPUTS ->
16433                              <1>       ;    none
16434                              <1>       ;
16435                              <1>       ;   ((Modified registers: ECX, ESI, EDI))
16436                              <1>       ;
16437                              <1>
16438                              <1>       ; 28/02/2017
16439                              <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
16440                              <1>       ;jna   short wswp
16441 000118D8 803D[DA030300]00    <1>       cmp    byte [u.fpsave], 0 ; 28/02/2017
16442 000118DF 7606                <1>       jna    short wswp
16443 000118E1 DD35[DC030300]      <1>       fnsave [u.fpregs] ; save floating point registers (94 bytes)
16444                              <1> wswp:
16445 000118E7 8B3D[B4030300]      <1>       mov    edi, [u.upage] ; process's user (u) structure page addr
16446 000118ED B938000000          <1>       mov    ecx, (U_SIZE + 3) / 4
16447 000118F2 BE[5C030300]        <1>       mov    esi, user ; active user (u) structure
16448 000118F7 F3A5                <1>       rep    movsd
16449                              <1>       ;
16450 000118F9 8B35[60030300]      <1>       mov    esi, [u.usp] ; esp (system stack pointer,
16451                              <1>                           ;      points to user registers)
16452 000118FF 8B0D[5C030300]      <1>       mov    ecx, [u.sp]  ; return address from the system call
16453                              <1>                           ; (for IRET)
16454                              <1>                           ; [u.sp] -> EIP (user)
16455                              <1>                           ; [u.sp+4]-> CS (user)
16456                              <1>                           ; [u.sp+8] -> EFLAGS (user)
16457                              <1>                           ; [u.sp+12] -> ESP (user)
16458                              <1>                           ; [u.sp+16] -> SS (user)
16459 00011905 29F1                <1>       sub    ecx, esi     ; required space for user registers
16460 00011907 83C114              <1>       add    ecx, 20           ; +5 dwords to return from system call
```

```
16461                                  <1>                             ; (for IRET)
16462 0001190A C1E902                  <1>       shr    ecx, 2
16463 0001190D F3A5                    <1>       rep    movsd
16464 0001190F C3                      <1>       retn
16465                                  <1>
16466                                  <1> rswap:  ; < swap in, swap from disk >
16467                                  <1>       ; 28/02/2017 (frstor)
16468                                  <1>       ; 15/01/2017
16469                                  <1>       ; 14/01/2017
16470                                  <1>       ; 21/05/2016
16471                                  <1>       ; 03/05/2016
16472                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16473                                  <1>       ; 09/05/2015 - 15/09/2015 (Retro UNIX 386 v1)
16474                                  <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
16475                                  <1>       ; 'rswap' reads a process whose number is in r1,
16476                                  <1>       ; from disk into core.
16477                                  <1>       ;
16478                                  <1>       ; Retro UNIX 386 v1 modification ->
16479                                  <1>       ;       User (u) structure content and the user's register content
16480                                  <1>       ;     will be restored from process's/user's UPAGE (a page for
16481                                  <1>       ;     saving 'u' structure and user registers for task switching).
16482                                  <1>       ;     u.usp - points to kernel stack address which contains
16483                                  <1>       ;             user's registers while entering system call.
16484                                  <1>       ;     u.sp  - points to kernel stack address
16485                                  <1>       ;             to return from system call -for IRET-.
16486                                  <1>       ;     [u.usp]+32+16 = [u.sp]
16487                                  <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
16488                                  <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
16489                                  <1>       ;
16490                                  <1>       ; RETRO UNIX 8086 v1 modification ->
16491                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
16492                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
16493                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
16494                                  <1>       ;     compatibles was using 1MB segmented memory
16495                                  <1>       ;     in 8086/8088 times.
16496                                  <1>       ;
16497                                  <1>       ; INPUTS ->
16498                                  <1>       ;    r1 - process number of process to be read in
16499                                  <1>       ;    p.break - negative of word count of process
16500                                  <1>       ;    p.dska - disk address of the process
16501                                  <1>       ;    u.emt - determines handling of emt's
16502                                  <1>       ;    u.ilgins - determines handling of illegal instructions
16503                                  <1>       ; OUTPUTS ->
16504                                  <1>       ;    8 = (u.ilgins)
16505                                  <1>       ;    24 = (u.emt)
16506                                  <1>       ;    swp - bit 10 is set to indicate read
16507                                  <1>       ;            (bit 15=0 when reading is done)
16508                                  <1>       ;    swp+2 - disk block address
16509                                  <1>       ;    swp+4 - negative word count
16510                                  <1>       ;      ((swp+6 - address of user structure))
16511                                  <1>       ;
16512                                  <1>       ; RETRO UNIX 8086 v1 input/output:
16513                                  <1>       ;
16514                                  <1>       ; INPUTS ->
16515                                  <1>       ;    AL      - new process number (to be swapped in)
16516                                  <1>       ; OUTPUTS ->
16517                                  <1>       ;    none
16518                                  <1>       ;
16519                                  <1>       ;   ((Modified registers: EAX, ECX, ESI, EDI, ESP))
16520                                  <1>       ;
16521                                  <1>       ; Retro UNIX 386 v1 - modification ! 14/05/2015
16522 00011910 89C6                    <1>       mov    esi, eax  ; process's user (u) structure page addr
16523 00011912 B938000000              <1>       mov    ecx, (U_SIZE + 3) / 4
16524 00011917 BF[5C030300]            <1>       mov    edi, user ; active user (u) structure
16525 0001191C F3A5                    <1>       rep    movsd
16526 0001191E 58                      <1>       pop    eax    ; 'rswap' return address
16527                                  <1>       ;
16528                                  <1>       ;cli
16529 0001191F 8B3D[60030300]          <1>       mov    edi, [u.usp] ; esp (system stack pointer,
16530                                  <1>                           ;     points to user registers)
16531 00011925 89FC                    <1>       mov    esp, edi     ; 14/01/2017
16532 00011927 8B0D[5C030300]          <1>       mov    ecx, [u.sp]  ; return address from the system call
16533                                  <1>                           ; (for IRET)
16534                                  <1>                           ; [u.sp] -> EIP (user)
16535                                  <1>                           ; [u.sp+4]-> CS (user)
16536                                  <1>                           ; [u.sp+8] -> EFLAGS (user)
16537                                  <1>                           ; [u.sp+12] -> ESP (user)
16538                                  <1>                           ; [u.sp+16] -> SS (user)
16539 0001192D 29F9                    <1>       sub    ecx, edi     ; required space for user registers
16540 0001192F 83C114                  <1>       add    ecx, 20           ; +5 dwords to return from system call
16541                                  <1>                           ; (for IRET)
16542 00011932 C1E902                  <1>       shr    ecx, 2
16543 00011935 F3A5                    <1>       rep    movsd
16544                                  <1>       ;mov   esp, [u.usp] ; 15/09/2015
16545                                  <1>       ;sti
16546                                  <1>       ; 28/02/2017
16547                                  <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
16548                                  <1>       ;jna   short rswp_retn
16549 00011937 803D[DA030300]00        <1>       cmp    byte [u.fpsave], 0
16550 0001193E 7606                    <1>       jna    short rswp_retn
16551 00011940 DD25[DC030300]          <1>       frstor [u.fpregs] ; restore floating point regs (94 bytes)
16552                                  <1> rswp_retn:
16553 00011946 50                      <1>       push   eax    ; 'rswap' return address
16554 00011947 C3                      <1>       retn
16555                                  <1>
16556                                  <1> putlu:
16557                                  <1>       ; 20/05/2016
16558                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16559                                  <1>       ; 10/05/2015 - 12/09/2015 (Retro UNIX 386 v1)
16560                                  <1>       ; 15/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
16561                                  <1>       ; 'putlu' is called with a process number in r1 and a pointer
16562                                  <1>       ; to lowest priority Q (runq+4) in r2. A link is created from
16563                                  <1>       ; the last process on the queue to process in r1 by putting
16564                                  <1>       ; the process number in r1 into the last process's link.
16565                                  <1>       ;
```

```
16566                                  <1>       ; INPUTS ->
16567                                  <1>       ;    r1 - user process number
16568                                  <1>       ;    r2 - points to lowest priority queue
16569                                  <1>       ;    p.dska - disk address of the process
16570                                  <1>       ;    u.emt - determines handling of emt's
16571                                  <1>       ;    u.ilgins - determines handling of illegal instructions
16572                                  <1>       ; OUTPUTS ->
16573                                  <1>       ;    r3 - process number of last process on the queue upon
16574                                  <1>       ;       entering putlu
16575                                  <1>       ;    p.link-1 + r3 - process number in r1
16576                                  <1>       ;    r2 - points to lowest priority queue
16577                                  <1>       ;
16578                                  <1>       ; ((Modified registers: EDX, EBX))
16579                                  <1>       ;
16580                                  <1>       ; / r1 = user process no.; r2 points to lowest priority queue
16581                                  <1>
16582                                  <1>       ; EBX = r2
16583                                  <1>       ; EAX = r1 (AL=r1b)
16584                                  <1>
16585                                  <1>       ; 20/05/2016
16586                                  <1>       ; AL = process number (1 to 16) // Retro UNIX 8086, 386 v1 //
16587                                  <1>       ;     (max. 16 processes available for current kernel version)
16588                                  <1>       ; EBX = run queue address ; 20/05/2016 (TRDOS 386)
16589                                  <1>             ; which is one of following addresses:
16590                                  <1>             ;  1) 'runq_event' high priority run queue
16591                                  <1>             ;  2) 'runq_normal' normal/regular priority run queue
16592                                  <1>             ;  3) 'runq_background' low priority run queue
16593                                  <1>
16594                                  <1>       ;mov   ebx, runq
16595 00011948 0FB613                 <1>       movzx        edx, byte [ebx]
16596 0001194B 43                     <1>       inc    ebx
16597 0001194C 20D2                   <1>       and    dl, dl
16598                                  <1>             ; tstb (r2)+ / is queue empty?
16599 0001194E 740A                   <1>             jz     short putlu_1
16600                                  <1>             ; beq 1f / yes, branch
16601 00011950 8A13                   <1>       mov    dl, [ebx] ; 12/09/2015
16602                                  <1>             ; movb (r2),r3 / no, save the "last user" process number
16603                                  <1>                         ; / in r3
16604 00011952 8882[9F000300]         <1>             mov    [edx+p.link-1], al
16605                                  <1>             ; movb r1,p.link-1(r3) / put pointer to user on
16606                                  <1>                         ; / "last users" link
16607 00011958 EB03                   <1>       jmp    short putlu_2
16608                                  <1>             ; br 2f /
16609                                  <1> putlu_1: ; 1:
16610 0001195A 8843FF                 <1>       mov    [ebx-1], al
16611                                  <1>                     ; movb r1,-1(r2) / user is only user;
16612                                  <1>                         ; / put process no. at beginning and at end
16613                                  <1> putlu_2: ; 2:
16614 0001195D 8803                   <1>       mov    [ebx], al
16615                                  <1>                     ; movb r1,(r2) / user process in r1 is now the last entry
16616                                  <1>                         ; / on the queue
16617 0001195F 88C2                   <1>       mov    dl, al
16618 00011961 88B2[9F000300]         <1>       mov    [edx+p.link-1], dh ; 0
16619                                  <1>             ; dec r2 / restore r2
16620 00011967 C3                     <1>       retn
16621                                  <1>             ; rts r0
16622                                  <1>
16623                                  <1> sysver:
16624                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16625 00011968 C705[64030300]0002-     <1>       mov    dword [u.r0], 200h ; AH = major version, AL = minor version
16625 00011970 0000                   <1>
16626 00011972 E955BEFFFF             <1>       jmp    sysret
16627                                  <1>
16628                                  <1>
16629                                  <1> syspri: ; change running priority (of the process)
16630                                  <1>       ; 21/05/2016
16631                                  <1>       ; 20/05/2026 - TRDOS 386 (TRDOS v2.0)
16632                                  <1>       ; INPUT ->
16633                                  <1>       ;     BL = priority level
16634                                  <1>       ;        0 = low running priority (running on background)
16635                                  <1>       ;        1 = normal/regular priority (running as regular)
16636                                  <1>       ;        2 = high/event priority (running for event)
16637                                  <1>       ;        >2 = invalid, it will accepted as 2 (event)
16638                                  <1>       ;        0FFh = get/return current running priority only
16639                                  <1>       ; OUTPUT ->
16640                                  <1>       ;     * if current [u.pri] < 2
16641                                  <1>       ;       if BL input < 0FFh ->
16642                                  <1>       ;          [u.pri] is updated as in BL input (0,1,2)
16643                                  <1>       ;       if BL input = 0FFh -> AL = [u.pri] (current)
16644                                  <1>       ;
16645                                  <1>       ;     * if current [u.pri] = 2
16646                                  <1>       ;       if BL input < 0FFh -> cf = 1 & AL = 2
16647                                  <1>       ;       if BL input = 0FFh -> cf = 0 & AL = 2
16648                                  <1>       ;
16649                                  <1>       ;     NOTE:
16650                                  <1>       ;     If [u.pri] = 2, it can not be changed to 1 or 0;
16651                                  <1>       ;     because, run queue of the running process is unspecified
16652                                  <1>       ;     at this      stage. Process might be started by a timer event
16653                                  <1>       ;     or priority might be changed to high by previous
16654                                  <1>       ;     'syspri' system     call. In both cases, the process is in
16655                                  <1>       ;     'runq_normal' or 'runq_background' queue.
16656                                  <1>       ;     As result of this fact, when the [u.quant] time quantum
16657                                  <1>       ;     of the process is elapsed or 'sysrele' system call is
16658                                  <1>       ;     instructed by the process, 'tswap' ('tswitch') procedure
16659                                  <1>       ;     will be called (to 'swap' or 'switch' out the procedure)
16660                                  <1>       ;     and it will not call 'putlu' to add the (stopping)
16661                                  <1>       ;     process to relevant run queue when [u.pri] = 2.
16662                                  <1>       ;     (Otherwise, it would be possible to add process to
16663                                  <1>       ;     a run queue while it is already in a run queue, wrongly.)
16664                                  <1>       ;
16665                                  <1>       ;     If [u.pri]< 2, 'tswap/tswitch' procedure will call
16666                                  <1>       ;     'putlu' to add process to relevant run queue
16667                                  <1>       ;     according to [u.pri] value. ('runq_normal' for 1,
16668                                  <1>       ;     'runq_background' for 0).
16669                                  <1>       ;
```

```
16670                                  <1>       ;     If BL input >= 2 and < 0FFh while [u.pri] < 2,
16671                                  <1>       ;     process will be added to 'runq_normal' queue and
16672                                  <1>       ;     [u.pri] will be set to 2. (in 'syspri' system call)
16673                                  <1>       ;
16674                                  <1>
16675 00011977 29C0                   <1>       sub   eax, eax ; 0
16676 00011979 A3[C8030300]           <1>       mov   [u.error], eax
16677                                  <1>
16678 0001197E A0[A9030300]           <1>       mov   al, [u.pri]
16679 00011983 A3[64030300]           <1>       mov   [u.r0], eax
16680                                  <1>
16681 00011988 FEC3                   <1>       inc   bl
16682 0001198A 0F843CBEFFFF           <1>       jz    sysret ; 0FFh -> 0, get priority level
16683                                  <1>
16684 00011990 3C02                   <1>       cmp   al, 2
16685 00011992 0F8314BEFFFF           <1>       jnb   error ; CF = 1 & AL = 2 (& last error = 0)
16686                                  <1>
16687 00011998 FECB                   <1>       dec   bl
16688 0001199A 80FB02                 <1>       cmp   bl, 2
16689 0001199D 7602                   <1>       jna   short syspri_1
16690 0001199F B302                   <1>       mov   bl, 2
16691                                  <1> syspri_1:
16692 000119A1 881D[A9030300]         <1>       mov   [u.pri], bl
16693 000119A7 80FB02                 <1>       cmp   bl, 2
16694 000119AA 0F821CBEFFFF           <1>       jb    sysret
16695                                  <1>
16696                                  <1>       ; here...
16697                                  <1>       ; Priority of current process has been changed to high
16698                                  <1>       ; ('run for event') but current process will be added to
16699                                  <1>       ; 'run as normal' queue. ('run for event' high priority
16700                                  <1>       ; queue is under control of timer -& RTC- interrupt only!)
16701                                  <1>       ;
16702                                  <1>       ; (Otherwise, process can fall into black hole!
16703                                  <1>       ; e.g. if it is not in waiting list and it has not got
16704                                  <1>       ; a timer event and it is not in a run queue!
16705                                  <1>       ; Because, when [u.pri] is 2, 'tswap/tswitch' will not
16706                                  <1>       ; add the stopping process to a run queue.)
16707                                  <1>
16708 000119B0 A0[B3030300]           <1>       mov   al, [u.uno]
16709 000119B5 BB[54030300]           <1>       mov   ebx, runq_normal ; normal priority !
16710                                  <1>                              ; [u.pri] is set to high
16711                                  <1>                              ; but 'runq_event' queue is set
16712                                  <1>                              ; only by the kernel's timer
16713                                  <1>                              ; event function (timer interrupt).
16714 000119BA E889FFFFFF             <1>       call  putlu
16715 000119BF E908BEFFFF             <1>       jmp   sysret
16716                                  <1>
16717                                  <1> cpass: ; / get next character from user area of core and put it in AL (r1)
16718                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
16719                                  <1>       ; 19/05/2015 - 18/10/2015 (Retro UNIX 386 v1)
16720                                  <1>       ; 14/08/2013 - 20/09/2013 (Retro UNIX 8086 v1)
16721                                  <1>       ; INPUTS ->
16722                                  <1>       ;     [u.base] = virtual address in user area
16723                                  <1>       ;     [u.count] = byte count (max.)
16724                                  <1>       ;     [u.pcount] = byte count in page (0 = reset)
16725                                  <1>       ; OUTPUTS ->
16726                                  <1>       ;     AL = the character which is pointed by [u.base]
16727                                  <1>       ;     zf = 1 -> transfer count has been completed
16728                                  <1>         ;
16729                                  <1>       ; ((Modified registers:  EAX, EDX, ECX))
16730                                  <1>       ;
16731 000119C4 833D[88030300]00       <1>       cmp   dword [u.count], 0  ; have all the characters been transferred
16732                                  <1>                                 ; i.e., u.count, # of chars. left
16733 000119CB 763F                   <1>       jna   short cpass_3    ; to be transferred = 0?) yes, branch
16734 000119CD FF0D[88030300]         <1>       dec   dword [u.count]         ; no, decrement u.count
16735                                  <1>         ; 19/05/2015
16736                                  <1>       ;(Retro UNIX 386 v1 - translation from user's virtual address
16737                                  <1>       ;                     to physical address
16738 000119D3 66833D[C4030300]00     <1>       cmp   word [u.pcount], 0 ; byte count in page = 0 (initial value)
16739                                  <1>                              ; 1-4095 --> use previous physical base address
16740                                  <1>                              ; in [u.pbase]
16741 000119DB 770E                   <1>       ja    short cpass_1
16742 000119DD 833D[BC030300]00       <1>       cmp   dword [u.ppgdir], 0 ; is the caller os kernel
16743 000119E4 7427                   <1>       je    short cpass_k       ; (sysexec, '/etc/init') ?  (MainProg)
16744 000119E6 E858FDFFFF             <1>       call  trans_addr_r
16745                                  <1> cpass_1:
16746 000119EB 66FF0D[C4030300]       <1>       dec   word [u.pcount]
16747                                  <1> cpass_2:
16748 000119F2 8B15[C0030300]         <1>       mov   edx, [u.pbase]
16749 000119F8 8A02                   <1>       mov   al, [edx]    ; take the character pointed to
16750                                  <1>                          ; by u.base and put it in r1
16751 000119FA FF05[8C030300]         <1>       inc   dword [u.nread] ; increment no. of bytes transferred
16752 00011A00 FF05[84030300]         <1>       inc   dword [u.base]  ; increment the buffer address to point to the
16753                                  <1>                               ; next byte
16754 00011A06 FF05[C0030300]         <1>       inc   dword [u.pbase]
16755                                  <1> cpass_3:
16756 00011A0C C3                     <1>       retn
16757                                  <1> cpass_k:
16758                                  <1>       ; 02/07/2015
16759                                  <1>       ; The caller is os kernel
16760                                  <1>       ; (get sysexec arguments from kernel's memory space)
16761 00011A0D 8B1D[84030300]         <1>       mov   ebx, [u.base]
16762 00011A13 66C705[C4030300]00-     <1>       mov   word [u.pcount], PAGE_SIZE ; 4096
16762 00011A1B 10                     <1>
16763 00011A1C 891D[C0030300]         <1>       mov   [u.pbase], ebx
16764 00011A22 EBCE                   <1>       jmp   short cpass_2
16765                                  <1>
16766                                  <1> transfer_to_user_buffer: ; fast transfer
16767                                  <1>       ; 27/05/2016
16768                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
16769                                  <1>       ;
16770                                  <1>       ; INPUT ->
16771                                  <1>       ;     ESI = source address in system space
16772                                  <1>       ;     EDI = user's buffer address
16773                                  <1>       ;     ECX = transfer (byte) count
```

```
16774                                  <1>       ;      [u.pgdir] = user's page directory
16775                                  <1>       ; OUTPUT ->
16776                                  <1>       ;      ECX = actual transfer count
16777                                  <1>       ;      cf = 1 -> error
16778                                  <1>       ;      [u.count] = remain byte count
16779                                  <1>       ;
16780                                  <1>       ; Modified registers: eax, ecx
16781                                  <1>       ;
16782                                  <1>
16783 00011A24 21C9                    <1>       and    ecx, ecx
16784 00011A26 743B                    <1>       jz     short ttub_4
16785                                  <1>
16786 00011A28 890D[88030300]          <1>       mov    [u.count], ecx
16787                                  <1>
16788 00011A2E 57                      <1>       push   edi
16789 00011A2F 56                      <1>       push   esi
16790 00011A30 53                      <1>       push   ebx
16791 00011A31 52                      <1>       push   edx
16792 00011A32 51                      <1>       push   ecx
16793                                  <1>
16794 00011A33 89FB                    <1>       mov    ebx, edi
16795 00011A35 81C300004000            <1>       add    ebx, CORE ; 27/05/2016
16796                                  <1> ttub_1:
16797                                  <1>       ; ebx = virtual (linear) address
16798                                  <1>       ; [u.pgdir] = user's page directory
16799 00011A3B E88247FFFF              <1>       call   get_physical_addr_x ; get physical address
16800 00011A40 7222                    <1>       jc     short ttub_5
16801                                  <1>       ; eax = physical address
16802                                  <1>       ; ecx = remain byte count in page (1-4096)
16803 00011A42 89C7                    <1>       mov    edi, eax
16804 00011A44 A1[88030300]            <1>       mov    eax, [u.count]
16805 00011A49 39C1                    <1>       cmp    ecx, eax
16806 00011A4B 7602                    <1>       jna    short ttub_2
16807 00011A4D 89C1                    <1>       mov    ecx, eax
16808                                  <1> ttub_2:
16809 00011A4F 29C8                    <1>       sub    eax, ecx
16810 00011A51 01CB                    <1>       add    ebx, ecx
16811 00011A53 F3A4                    <1>       rep    movsb
16812 00011A55 A3[88030300]            <1>       mov    [u.count], eax
16813 00011A5A 09C0                    <1>       or     eax, eax
16814 00011A5C 75DD                    <1>       jnz    short ttub_1
16815                                  <1> ttub_retn:
16816                                  <1> tfub_retn:
16817 00011A5E 59                      <1>       pop    ecx ; transfer count = actual transfer count
16818                                  <1> ttub_3:
16819 00011A5F 5A                      <1>       pop    edx
16820 00011A60 5B                      <1>       pop    ebx
16821 00011A61 5E                      <1>       pop    esi
16822 00011A62 5F                      <1>       pop    edi
16823                                  <1> ttub_4:
16824 00011A63 C3                      <1>       retn
16825                                  <1> ttub_5:
16826 00011A64 59                      <1>       pop    ecx
16827 00011A65 2B0D[88030300]          <1>       sub    ecx, [u.count] ; actual transfer count
16828 00011A6B F9                      <1>       stc
16829 00011A6C EBF1                    <1>       jmp    short ttub_3
16830                                  <1>
16831                                  <1> transfer_from_user_buffer: ; fast transfer
16832                                  <1>       ; 27/05/2016
16833                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
16834                                  <1>       ;
16835                                  <1>       ; INPUT ->
16836                                  <1>       ;      ESI = user's buffer address
16837                                  <1>       ;      EDI = destination address in system space
16838                                  <1>       ;      ECX = transfer (byte) count
16839                                  <1>       ;      [u.pgdir] = user's page directory
16840                                  <1>       ; OUTPUT ->
16841                                  <1>       ;      ecx = actual transfer count
16842                                  <1>       ;      cf = 1 -> error
16843                                  <1>       ;      [u.count] = remain byte count
16844                                  <1>       ;
16845                                  <1>       ; Modified registers: eax, ecx
16846                                  <1>       ;
16847                                  <1>
16848 00011A6E 21C9                    <1>       and    ecx, ecx
16849                                  <1>       ;jz    short tfub_4
16850 00011A70 74F1                    <1>       jz     short ttub_4
16851                                  <1>
16852 00011A72 890D[88030300]          <1>       mov    [u.count], ecx
16853                                  <1>
16854 00011A78 57                      <1>       push   edi
16855 00011A79 56                      <1>       push   esi
16856 00011A7A 53                      <1>       push   ebx
16857 00011A7B 52                      <1>       push   edx
16858 00011A7C 51                      <1>       push   ecx
16859                                  <1>
16860 00011A7D 89F3                    <1>       mov    ebx, esi
16861 00011A7F 81C300004000            <1>       add    ebx, CORE ; 27/05/2016
16862                                  <1> tfub_1:
16863                                  <1>       ; ebx = virtual (linear) address
16864                                  <1>       ; [u.pgdir] = user's page directory
16865 00011A85 E83847FFFF              <1>       call   get_physical_addr_x ; get physical address
16866                                  <1>       ;jc    short tfub_5
16867 00011A8A 72D8                    <1>       jc     short ttub_5
16868                                  <1>       ; eax = physical address
16869                                  <1>       ; ecx = remain byte count in page (1-4096)
16870 00011A8C 89C6                    <1>       mov    esi, eax
16871 00011A8E A1[88030300]            <1>       mov    eax, [u.count]
16872 00011A93 39C1                    <1>       cmp    ecx, eax
16873 00011A95 7602                    <1>       jna    short tfub_2
16874 00011A97 89C1                    <1>       mov    ecx, eax
16875                                  <1> tfub_2:
16876 00011A99 29C8                    <1>       sub    eax, ecx
16877 00011A9B 01CB                    <1>       add    ebx, ecx
16878 00011A9D F3A4                    <1>       rep    movsb
```

```
16879 00011A9F A3[88030300]       <1>       mov    [u.count], eax
16880 00011AA4 09C0               <1>       or     eax, eax
16881 00011AA6 75DD               <1>       jnz    short tfub_1
16882                             <1>
16883 00011AA8 EBB4               <1>       jmp    short tfub_retn
16884                             <1>
16885                             <1> ;tfub_retn:
16886                             <1> ;     pop    ecx ; transfer count = actual transfer count
16887                             <1> ;tfub_3:
16888                             <1> ;     pop    edx
16889                             <1> ;     pop    ebx
16890                             <1> ;     pop    esi
16891                             <1> ;     pop    edi
16892                             <1> ;tfub_4:
16893                             <1> ;     retn
16894                             <1> ;tfub_5:
16895                             <1> ;     pop    ecx
16896                             <1> ;     sub    ecx, [u.count] ; actual transfer count
16897                             <1> ;     stc
16898                             <1> ;     jmp    short tfub_3
16899                             <1>
16900                             <1> sysfff: ; <Find First File>
16901                             <1>       ; 17/10/2016
16902                             <1>       ; 16/10/2016
16903                             <1>       ; 15/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
16904                             <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
16905                             <1>       ;            ("loc_INT21h_find_first_file")
16906                             <1>       ; TRDOS 8086 (v1.0)
16907                             <1>         ;   07/08/2011
16908                             <1>         ;   Find First File
16909                             <1>       ;     INPUT:
16910                             <1>         ;       CX= Attributes
16911                             <1>         ;       DS:DX= Pointer to filename
16912                             <1>       ;     MSDOS OUTPUT:
16913                             <1>       ;         DTA: (Default address: PSP offset 80h)
16914                             <1>       ;         Offset  Descrription
16915                             <1>       ;         0     Reserved for use find next file
16916                             <1>       ;         21    Attribute of file found
16917                             <1>       ;         22    Time stamp of file
16918                             <1>       ;         24    Date stamp of file
16919                             <1>       ;         26    File size in bytes
16920                             <1>       ;         30    Filename and extension (zero terminated)
16921                             <1>       ;     If cf = 1:
16922                             <1>       ;         Error Codes: (in AX)
16923                             <1>       ;           2 - File not found
16924                             <1>       ;           18 - No more files
16925                             <1>       ;
16926                             <1>       ; TRDOS 386 (v2.0)
16927                             <1>       ; 15/10/2016
16928                             <1>       ;
16929                             <1>       ; INPUT ->
16930                             <1>       ;       CL = File attributes
16931                             <1>       ;          bit 0 (1) - Read only file (R)
16932                             <1>       ;          bit 1 (1) - Hidden file (H)
16933                             <1>       ;          bit 2 (1) - System file (R)
16934                             <1>       ;          bit 3 (1) - Volume label/name (V)
16935                             <1>       ;          bit 4 (1) - Subdirectory (D)
16936                             <1>       ;          bit 5 (1) - File has been archived (A)
16937                             <1>       ;       CH = 0 -> Return basic parameters (24 bytes)
16938                             <1>       ;       CH > 0 -> Return FindFile structure/table (128 bytes)
16939                             <1>       ;         EBX = Pointer to filename (ASCIIZ) -path-
16940                             <1>       ;       EDX = File parameters buffer address
16941                             <1>       ;           (buffer size = 24 bytes if CH input = 0)
16942                             <1>       ;           (buffer size = 128 bytes if CH input > 0)
16943                             <1>       ;
16944                             <1>       ; OUTPUT ->
16945                             <1>       ;       EAX = 0 if CH input > 0
16946                             <1>       ;       EAX = First cluster number of file if CH input = 0
16947                             <1>       ;       EDX = File parameters table/structure address
16948                             <1>       ;       Basic Parameters:
16949                             <1>       ;           Offset  Description
16950                             <1>       ;           ------ ---------------
16951                             <1>       ;           0      File Attributes
16952                             <1>       ;           1      Ambiguous filename chars are used sign
16953                             <1>       ;                  (0 = filename fits exactly with request)
16954                             <1>       ;                  (>0 = ambiguous filename chars are used)
16955                             <1>       ;           2      Time stamp of file
16956                             <1>       ;           4      Date stamp of file
16957                             <1>       ;           6      File size in bytes
16958                             <1>       ;           10     Short Filename (ASCIIZ, max. 13 bytes)
16959                             <1>       ;           23     Longname Length (1-255) if existing
16960                             <1>       ;
16961                             <1>       ;         cf = 1 -> Error code in AL
16962                             <1>       ;
16963                             <1>       ; Modified Registers: EAX (at the return of system call)
16964                             <1>       ;
16965                             <1>       ; TR-DOS FindFile (FFF) Structure (128 bytes):
16966                             <1>       ; 09/10/2011 (DIR.ASM) - 10/02/2016 (trdoskx.s)
16967                             <1>       ;
16968                             <1>       ; Offset     Parameter           Size
16969                             <1>       ; ------     ------------------  --------
16970                             <1>       ; 0          FindFile_Drv        1 byte
16971                             <1>       ; 1          FindFile_Directory  65 bytes
16972                             <1>       ; 66         FindFile_Name       13 bytes
16973                             <1>       ; 79         FindFile_LongNameEntryLength 1 byte
16974                             <1>       ;Above 80 bytes form
16975                             <1>       ;TR-DOS Source/Destination File FullName Format/Structure
16976                             <1>       ; 80         FindFile_AttributesMask 1 word
16977                             <1>       ; 82         FindFile_DirEntry   32 bytes (*)
16978                             <1>       ; 114        FindFile_DirFirstCluster 1 double word
16979                             <1>       ; 118        FindFile_DirCluster 1 double word
16980                             <1>       ; 122        FindFile_DirEntryNumber 1 word
16981                             <1>       ; 124        FindFile_MatchCounter     1 word
16982                             <1>       ; 126        FindFile_Reserved   1 word
16983                             <1>       ; (*) MS-DOS, FAT 12-16-32 classic directory entry (32 bytes)
```

```
16984                                  <1>
16985                                  <1>       ;mov   [u.namep], ebx
16986                                  <1>       ; 16/10/2016
16987 00011AAA 8915[A0950100]          <1>       mov    [FFF_UBuffer], edx
16988 00011AB0 66890D[A5950100]        <1>       mov    [FFF_Attrib], cx ; [FFF_RType] = ch
16989                                  <1>                   ; Attributes in CL, return data type in CH
16990 00011AB7 89DE                    <1>       mov    esi, ebx
16991                                  <1>       ; file name is forced, change directory as temporary
16992                                  <1>       ;mov   ax, 1
16993                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
16994                                  <1>       ;call  set_working_path
16995 00011AB9 E8E2130000              <1>       call   set_working_path_x ; 17/10/2016
16996 00011ABE 731D                    <1>       jnc    short sysfff_0
16997                                  <1>
16998 00011AC0 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
16999 00011AC2 7505                    <1>       jnz    short sysfff_err
17000                                  <1>
17001                                  <1>       ; eax = 0
17002 00011AC4 B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
17003                                  <1> sysfff_err:
17004 00011AC9 A3[64030300]            <1>       mov    [u.r0], eax
17005 00011ACE A3[C8030300]            <1>       mov    [u.error], eax
17006 00011AD3 E89D140000              <1>         call       reset_working_path
17007 00011AD8 E9CFBCFFFF              <1>       jmp    error
17008                                  <1>
17009                                  <1> sysfff_0:
17010                                  <1>       ;sub   ah, ah ; ah = 0
17011 00011ADD 8A0424                  <1>       mov    al, [esp]
17012 00011AE0 08C0                    <1>       or     al, al
17013 00011AE2 7412                    <1>       jz     short sysfff_2
17014 00011AE4 B410                    <1>       mov    ah, 10h
17015 00011AE6 A808                    <1>       test   al, 08h
17016 00011AE8 7503                    <1>       jnz    short sysfff_1
17017 00011AEA 80CC08                  <1>       or     ah, 08h
17018                                  <1> sysfff_1:
17019 00011AED 2410                    <1>       and    al, 10h ; Directory
17020 00011AEF 7405                    <1>       jz     short sysfff_2
17021 00011AF1 80E408                  <1>       and    ah, 08h
17022 00011AF4 30C0                    <1>       xor    al, al ; When a directory is searched,
17023                                  <1>                     ; filename will be returned even if
17024                                  <1>                     ; it is not a directory!
17025                                  <1>                     ; Because: (in order to prevent
17026                                  <1>                     ; creating a dir with existing file name)
17027                                  <1>                     ; Dir and file names must not be same!
17028                                  <1>                     ; (return attribute must be checked)
17029                                  <1> sysfff_2:
17030                                  <1>       ; AX = Attributes mask
17031                                  <1>             ; AL = AND mask (result must be equal to AL)
17032                                  <1>             ; AH = Negative AND mask (result must be ZERO)
17033                                  <1>       ; ESI = FindFile_Name address
17034                                  <1>
17035 00011AF6 E85678FFFF              <1>       call   find_first_file
17036 00011AFB 72CC                    <1>       jc     short sysfff_err ; eax = 2 (File not found !)
17037                                  <1>
17038                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
17039                                  <1>       ; EDI = Directory Buffer Directory Entry Location
17040                                  <1>       ; EAX = File Size
17041                                  <1>       ;  BL = Attributes of The File/Directory
17042                                  <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
17043                                  <1>       ;  DX > 0 : Ambiguous filename chars are used
17044                                  <1>
17045                                  <1> sysfff_3:
17046                                  <1>       ; 16/10/2016
17047 00011AFD 668B0D[A5950100]        <1>       mov    cx, [FFF_Attrib]
17048                                  <1>       ; Attribs in CL, return data type in CH
17049                                  <1>
17050                                  <1>       ;or    cl, cl
17051                                  <1>       ;jz    short sysfff_4 ; 0 = No filter
17052 00011B04 80F1FF                  <1>       xor    cl, 0FFh
17053 00011B07 20D9                    <1>       and    cl, bl
17054 00011B09 7409                    <1>       jz     short sysfff_4
17055                                  <1>
17056                                  <1>       ;mov   eax, 2 ; 'file not found !' error
17057                                  <1>       ;jmp   short sysfff_err_1
17058                                  <1>
17059                                  <1>       ; 16/10/2016
17060 00011B0B E8F078FFFF              <1>       call   find_next_file
17061 00011B10 72B7                    <1>       jc     short sysfff_err ; eax = 12 (no more files !)
17062 00011B12 EBE9                    <1>       jmp    short sysfff_3
17063                                  <1>
17064                                  <1> sysfff_4:
17065 00011B14 20ED                    <1>       and    ch, ch ; [FFF_RType]
17066 00011B16 7412                    <1>       jz     short sysfff_5
17067 00011B18 B980000000              <1>       mov    ecx, 128 ; ; transfer length
17068 00011B1D 880D[A4950100]          <1>       mov    [FFF_Valid], cl
17069                                  <1> sysfnf_11:
17070 00011B23 BE[56920100]            <1>       mov    esi, FindFile_Drv
17071 00011B28 EB44                    <1>       jmp    short sysfff_6
17072                                  <1> sysfff_5:
17073                                  <1>       ;mov   esi, FindFile_DirEntry
17074 00011B2A B918000000              <1>       mov    ecx, 24  ; transfer length
17075 00011B2F 880D[A4950100]          <1>       mov    [FFF_Valid], cl
17076                                  <1> sysfnf_12:
17077 00011B35 BF[609A0100]            <1>       mov    edi, DTA ; FFF data transfer address
17078                                  <1>       ;mov   al, [esi+DirEntry_Attr] ; 11
17079 00011B3A 88D8                    <1>       mov    al, bl ; File/Dir Attributes
17080 00011B3C 887F17                  <1>       mov    [edi+23], bh ; Longname length (0= none)
17081 00011B3F AA                      <1>       stosb
17082 00011B40 88D0                    <1>       mov    al, dl ; DL is for '?'
17083 00011B42 00F0                    <1>       add    al, dh ; DH is for '*'
17084                                  <1>       ; AL > 0 if ambiguous file name wildcards are used
17085 00011B44 AA                      <1>       stosb
17086 00011B45 8B4616                  <1>       mov    eax, [esi+DirEntry_WrtTime] ; 22
17087 00011B48 AB                      <1>         stosd      ; DirEntry_WrtTime & DirEntry_WrtDate
17088 00011B49 8B461C                  <1>         mov  eax, [esi+DirEntry_FileSize] ; 28
```

```
17089 00011B4C AB                 <1>         stosd
17090 00011B4D 668B4614           <1>       mov   ax, [esi+DirEntry_FstClusHI] ; 20
17091 00011B51 66C1E010           <1>       shl   ax, 16
17092 00011B55 668B461A           <1>       mov   ax, [esi+DirEntry_FstClusLO] ; 26
17093 00011B59 A3[64030300]       <1>       mov   [u.r0], eax ; First Cluster
17094                             <1>
17095                             <1>      ;mov   esi, FindFile_DirEntry
17096 00011B5E E84F140000         <1>       call  get_file_name
17097                             <1>
17098 00011B63 8A0D[A4950100]     <1>       mov   cl, [FFF_Valid]
17099 00011B69 BE[609A0100]       <1>       mov   esi, DTA ; FFF data transfer address
17100                             <1> sysfff_6:
17101 00011B6E 8B3D[A0950100]     <1>       mov   edi, [FFF_UBuffer] ; user's buffer address (edx)
17102 00011B74 E8ABFEFFFF         <1>       call  transfer_to_user_buffer
17103                             <1>
17104 00011B79 890D[64030300]     <1>       mov   [u.r0], ecx ; actual transfer count
17105 00011B7F E8F1130000         <1>       call  reset_working_path
17106 00011B84 E943BCFFFF         <1>       jmp   sysret
17107                             <1>
17108                             <1> sysfnf: ; <Find Next File>
17109                             <1>       ; 16/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
17110                             <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
17111                             <1>       ;            ("loc_INT21h_find_next_file")
17112                             <1>       ; TRDOS 8086 (v1.0)
17113                             <1>       ;     07/08/2011
17114                             <1>       ;     Find First File
17115                             <1>       ;     INPUT:
17116                             <1>       ;         none
17117                             <1>       ;     MSDOS OUTPUT:
17118                             <1>       ;         DTA: (Default address: PSP offset 80h)
17119                             <1>       ;         Offset  Descrription
17120                             <1>       ;         0     Reserved for use find next file
17121                             <1>       ;         21    Attribute of file found
17122                             <1>       ;         22    Time stamp of file
17123                             <1>       ;         24    Date stamp of file
17124                             <1>       ;         26    File size in bytes
17125                             <1>       ;         30    Filename and extension (zero terminated)
17126                             <1>       ;     If cf = 1:
17127                             <1>       ;         Error Codes: (in AX)
17128                             <1>       ;            18 - No more files
17129                             <1>       ;
17130                             <1>       ; TRDOS 386 (v2.0)
17131                             <1>       ; 16/10/2016
17132                             <1>       ;
17133                             <1>       ; INPUT ->
17134                             <1>       ;        none
17135                             <1>       ; OUTPUT ->
17136                             <1>       ;        EAX = 0 if CH input of 'Find First File' > 0
17137                             <1>       ;        EAX = First cluster number of file
17138                             <1>       ;             if CH input of 'Find First File' = 0
17139                             <1>       ;        EDX = File parameters table/structure address
17140                             <1>       ;
17141                             <1>       ;         cf = 1 -> Error code in AL
17142                             <1>       ;
17143                             <1>       ; Modified Registers: EAX (at the return of system call)
17144                             <1>
17145                             <1>       ;
17146                             <1>       ; Note: If byte [FFF_Valid] = 0
17147                             <1>       ;     'sysfnf' will return with 'no more files' error.
17148                             <1>       ;     If byte [FFF_Valid] = 24
17149                             <1>       ;     'sysfnf' will return with 32 bytes basic parameters
17150                             <1>       ;     at the address which is in EDX.
17151                             <1>       ;     If byte [FFF_Valid] = 128
17152                             <1>       ;     'sysfnf' will return with 128 bytes Find File
17153                             <1>       ;     Structure/Table at the address which is in EDX.
17154                             <1>
17155 00011B89 803D[A4950100]00   <1>       cmp   byte [FFF_Valid], 0
17156 00011B90 7714               <1>       ja    short stsfnf_0
17157                             <1>       ; 'no more files !' error
17158 00011B92 B80C000000         <1>       mov   eax, ERR_NO_MORE_FILES ; 12
17159 00011B97 A3[64030300]       <1>       mov   [u.r0], eax
17160 00011B9C A3[C8030300]       <1>       mov   [u.error], eax
17161 00011BA1 E906BCFFFF         <1>       jmp   error
17162                             <1> stsfnf_0:
17163                             <1>       ;cmp  byte [FFF_Valid], 128
17164                             <1>       ;je   short stsfnf_1
17165                             <1>       ;cmp  byte [FFF_Valid], 24
17166                             <1>       ;je   short stsfnf_1
17167                             <1>       ;mov  [FFF_Valid], 24 ; Default
17168                             <1> stsfnf_1:
17169 00011BA6 0FB61D[B6890100]   <1>       movzx ebx, byte [Current_Drv]
17170 00011BAD 66891D[AA950100]   <1>       mov   [SWP_DRV], bx
17171 00011BB4 8A15[56920100]     <1>       mov   dl, [FindFile_Drv]
17172 00011BBA 38DA               <1>       cmp   dl, bl
17173 00011BBC 750B               <1>       jne   short stsfnf_2
17174 00011BBE 86FB               <1>       xchg  bh, bl
17175 00011BC0 BE00010900         <1>       mov   esi, Logical_DOSDisks
17176 00011BC5 01DE               <1>       add   esi, ebx
17177 00011BC7 EB0D               <1>       jmp   short sysfnf_3
17178                             <1>
17179                             <1> stsfnf_2:
17180 00011BC9 FE05[AB950100]     <1>       inc   byte [SWP_DRV_chg]
17181                             <1>
17182 00011BCF E8DA63FFFF         <1>       call  change_current_drive
17183 00011BD4 7245               <1>       jc    short sysfnf_err_1 ; read error !
17184                             <1>                                ; (do not stop, because
17185                             <1>                                ; we don't have a
17186                             <1>                                ; 'no more files'
17187                             <1>                                ; -file not found- error,
17188                             <1>                                ; next sysfnf system call
17189                             <1>                                ; may solve the problem,
17190                             <1>                                ; after re-placing the disk)
17191                             <1> sysfnf_3:
17192 00011BD6 A1[CC920100]       <1>       mov   eax, [FindFile_DirCluster]
17193 00011BDB 21C0               <1>       and   eax, eax
```

```
17194 00011BDD 7550               <1>       jnz    short sysfnf_6
17195                             <1>
17196 00011BDF 803D[B5890100]02   <1>       cmp    byte [Current_FATType], 2
17197 00011BE6 772C               <1>       ja     short sysfnf_err_0 ; invalid, we neeed to stop !?
17198 00011BE8 803D[B5890100]01   <1>       cmp    byte [Current_FATType], 1
17199 00011BEF 7223               <1>       jb     short sysfnf_err_0 ; invalid, we neeed to stop !?
17200                             <1>
17201 00011BF1 3805[DC900100]     <1>       cmp    byte [DirBuff_ValidData], al ; 0
17202 00011BF7 7608               <1>       jna    short sysfnf_4
17203                             <1>
17204 00011BF9 3B05[E1900100]     <1>       cmp    eax, [DirBuff_Cluster] ; 0 ?
17205 00011BFF 745E               <1>       je     short sysfnf_9
17206                             <1>
17207                             <1>       ;cmp   byte [Current_Dir_Level], 0
17208                             <1>       ;ja    short sysfnf_4
17209                             <1>       ;jna   short sysfnf_9
17210                             <1>
17211                             <1> sysfnf_4:
17212 00011C01 FE05[AB950100]     <1>       inc    byte [SWP_DRV_chg]
17213 00011C07 E885B1FFFF         <1>       call   load_FAT_root_directory
17214 00011C0C 7351               <1>       jnc    short sysfnf_9
17215                             <1>       ; eax = error code (17, 'drv not ready or read error')
17216 00011C0E EB0B               <1>       jmp    short sysfnf_err_1 ; read error ! (no FNF stop)
17217                             <1>                                 ; (if you want, try again,
17218                             <1>                                 ;  after re-placing the disk)
17219                             <1> sysfnf_5:
17220 00011C10 3C0C               <1>       cmp    al, 12 ; 'no more files' error
17221 00011C12 7507               <1>       jne    short sysfnf_err_1 ; (no FNF stop -sysfnf will try
17222                             <1>                                 ;  to read the directory again,
17223                             <1>                                 ;  if the user calls sysfnf
17224                             <1>                                 ;  just after this error return-)
17225                             <1>       ; (FNF stop -sysfnf will not try
17226                             <1>       ;  to read the directory again-)
17227                             <1>
17228                             <1> sysfnf_err_0:
17229 00011C14 C605[A4950100]00   <1>       mov    byte [FFF_Valid], 0 ; FNF stop sign
17230                             <1> sysfnf_err_1:
17231 00011C1B A3[64030300]       <1>       mov    [u.r0], eax
17232 00011C20 A3[C8030300]       <1>       mov    [u.error], eax
17233 00011C25 E84B130000         <1>       call   reset_working_path
17234 00011C2A E97DBBFFFF         <1>       jmp    error
17235                             <1>
17236                             <1> sysfnf_6:
17237 00011C2F 803D[DC900100]00   <1>       cmp    byte [DirBuff_ValidData], 0
17238 00011C36 7608               <1>       jna    short sysfnf_7
17239                             <1>
17240 00011C38 3B05[E1900100]     <1>       cmp    eax, [DirBuff_Cluster]
17241 00011C3E 741F               <1>       je     short sysfnf_9
17242                             <1>
17243                             <1> sysfnf_7:
17244 00011C40 FE05[AB950100]     <1>       inc    byte [SWP_DRV_chg]
17245 00011C46 803D[B5890100]01   <1>       cmp    byte [Current_FATType], 1
17246 00011C4D 7309               <1>       jnb    short sysfnf_8
17247                             <1>
17248                             <1>       ; Singlix (TRFS) File System
17249                             <1>       ; (access via compatibility buffer)
17250 00011C4F E805B2FFFF         <1>       call   load_FS_sub_directory
17251 00011C54 7309               <1>       jnc    short sysfnf_9
17252                             <1>
17253 00011C56 EBC3               <1>       jmp    short sysfnf_err_1 ; read error (no FNF stop)
17254                             <1>
17255                             <1> sysfnf_8:
17256 00011C58 E8BFB1FFFF         <1>       call   load_FAT_sub_directory
17257 00011C5D 72BC               <1>       jc     short sysfnf_err_1 ; read error (no FNF stop)
17258                             <1>
17259                             <1> sysfnf_9:
17260 00011C5F E89C77FFFF         <1>       call   find_next_file
17261 00011C64 72AA               <1>       jc     short sysfnf_5
17262                             <1>
17263 00011C66 A0[A5950100]       <1>       mov    al, [FFF_Attrib]
17264                             <1>       ;or    al, al
17265                             <1>       ;jz    short sysfnf_10 ; 0 = No filter
17266 00011C6B 34FF               <1>       xor    al, 0FFh
17267 00011C6D 20D8               <1>       and    al, bl
17268 00011C6F 75EE               <1>       jnz    short sysfnf_9 ; search for next file until
17269                             <1>                               ; an error return from
17270                             <1>                               ; find_next_file procedure
17271                             <1> sysfnf_10:
17272 00011C71 0FB60D[A4950100]   <1>       movzx  ecx, byte [FFF_Valid]
17273 00011C78 80F980             <1>       cmp    cl, 128 ; complete FindFile structure/table
17274 00011C7B 0F84A2FEFFFF       <1>       je     sysfnf_11
17275                             <1>       ;cmp   cl, 24  ; basic parameters
17276                             <1>       ;je    sysfnf_12
17277 00011C81 E9AFFEFFFF         <1>       jmp    sysfnf_12
17278                             <1>
17279                             <1> writei:
17280                             <1>       ; 26/10/2016
17281                             <1>       ; 25/10/2016
17282                             <1>       ; 23/10/2016
17283                             <1>       ; 22/10/2016
17284                             <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
17285                             <1>       ; 19/05/2015 - 20/05/2015 (Retro UNIX 386 v1)
17286                             <1>       ; 12/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
17287                             <1>       ;
17288                             <1>       ; Write data to file with first cluster number in EAX
17289                             <1>       ;
17290                             <1>       ; INPUTS ->
17291                             <1>       ;    EAX - First cluster number of the file
17292                             <1>       ;    EBX - File number (Open file index number)
17293                             <1>       ;    u.count - byte count to be written
17294                             <1>       ;    u.base - points to user buffer
17295                             <1>       ;    u.fofp - points to dword with current file offset
17296                             <1>       ;    i.size - file size
17297                             <1>       ;    cdev - logical dos drive number of the file
17298                             <1>       ; OUTPUTS ->
```

```
17299                                  <1>       ;    u.count - cleared
17300                                  <1>       ;    u.nread - accumulates total bytes passed back
17301                                  <1>       ;    i.size - new file size (if file byte offset overs file size)
17302                                  <1>       ;    u.fofp - points to u.off (with new offset value)
17303                                  <1>       ;
17304                                  <1>       ; (Retro UNIX Prototype : 11/11/2012 - 18/11/2012, UNIXCOPY.ASM)
17305                                  <1>       ; ((Modified registers: eax, edx, ebx, ecx, esi, edi, ebp))
17306                                  <1>
17307 00011C86 31C9                   <1>       xor    ecx, ecx
17308 00011C88 890D[8C030300]         <1>       mov    [u.nread], ecx  ; 0
17309 00011C8E 66890D[C4030300]       <1>       mov    [u.pcount], cx ; 19/05/2015
17310 00011C95 390D[88030300]         <1>       cmp    [u.count], ecx
17311 00011C9B 7701                   <1>       ja     short writei_1
17312 00011C9D C3                     <1>       retn
17313                                  <1> writei_1:
17314 00011C9E 881D[64950100]         <1>       mov    [writei.ofn], bl ; Open file number
17315 00011CA4 880D[9F950100]         <1>       mov    [setfmod], cl ; 0 ; reset 'update lm date&time' sign
17316                                  <1> dskw_0:
17317                                  <1>       ; 26/10/2016
17318                                  <1>       ; 22/10/2016, 23/10/2016, 25/10/2016
17319                                  <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
17320                                  <1>       ; 31/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
17321                                  <1>       ; 26/04/2013 - 20/09/2013 (Retro UNIX 8086 v1)
17322                                  <1>       ;
17323                                  <1>       ; 01/08/2013 (mkdir_w check)
17324 00011CAA E8D7000000             <1>       call   mget_w
17325                                  <1>       ; eax = sector/block number
17326                                  <1>
17327 00011CAF 8B1D[74030300]         <1>       mov     ebx, [u.fofp]
17328 00011CB5 8B13                   <1>       mov    edx, [ebx]
17329 00011CB7 81E2FF010000           <1>       and    edx, 1FFh  ; / test the lower 9 bits of the file offset
17330 00011CBD 750C                   <1>       jnz    short dskw_1 ; / if its non-zero, branch
17331                                  <1>                           ; if zero, file offset = 0,
17332                                  <1>                                 ; / 512, 1024,...(i.e., start of new block)
17333 00011CBF 813D[88030300]0002-     <1>       cmp    dword [u.count], 512
17333 00011CC7 0000                   <1>
17334                                  <1>                               ; / if zero, is there enough data to fill
17335                                  <1>                               ; / an entire block? (i.e., no. of
17336 00011CC9 7337                   <1>       jnb    short dskw_2 ; / bytes to be written greater than 512.?
17337                                  <1>                             ; / Yes, branch. Don't have to read block
17338                                  <1> dskw_1: ; in as no past info. is to be saved
17339                                  <1>       ; (the entire block will be overwritten).
17340                                  <1>       ; 23/10/2016
17341                                  <1>
17342 00011CCB BB[94070300]           <1>       mov    ebx, writei_buffer
17343                                  <1>       ; esi = logical dos drive description table address
17344                                  <1>       ; eax = sector number
17345                                  <1>       ; ebx = buffer address (in kernel's memory space)
17346                                  <1>       ; ecx = sector count
17347 00011CD0 B901000000             <1>       mov    ecx, 1
17348 00011CD5 E8A30D0000             <1>       call   disk_read
17349                                  <1>       ;call  dskrd  ; / no, must retain old info..
17350                                  <1>                               ; / Hence, read block 'r1' into an I/O buffer
17351 00011CDA 7326                   <1>       jnc    short dskw_2
17352                                  <1>
17353                                  <1>       ; disk read error
17354 00011CDC B811000000             <1>       mov    eax, 17 ; drive not ready or READ ERROR !
17355                                  <1> dskw_err: ; jump from disk write error
17356 00011CE1 A3[64030300]           <1>       mov    [u.r0], eax
17357 00011CE6 A3[C8030300]           <1>       mov    [u.error], eax
17358                                  <1>
17359 00011CEB 803D[9F950100]00       <1>       cmp    byte [setfmod], 0
17360 00011CF2 0F86B4BAFFFF           <1>       jna    error
17361                                  <1>
17362 00011CF8 E8AF030000             <1>       call   update_file_lmdt ; update last modif. date&time of the file
17363                                  <1>       ;mov   byte [setfmod], 0
17364                                  <1>
17365 00011CFD E9AABAFFFF             <1>       jmp    error
17366                                  <1>
17367                                  <1> dskw_2: ; 3:
17368                                  <1>       ; 23/10/2016
17369 00011D02 C605[40950100]01       <1>       mov    byte [writei.valid], 1 ; writei buffer contains valid data
17370 00011D09 56                     <1>       push   esi ; logical dos drive description table address
17371                                  <1>       ; EAX (r1) = block/sector number
17372                                  <1>       ;call  wslot
17373                                  <1>              ; jsr r0,wslot / set write and inhibit bits in I/O queue,
17374                                  <1>                       ; / proc. status=0, r5 points to 1st word of data
17375 00011D0A 803D[C6030300]00       <1>       cmp    byte [u.kcall], 0
17376 00011D11 770F                   <1>       ja     short dskw_4 ; zf=0 -> the caller is 'mkdir'
17377                                  <1>       ;
17378 00011D13 66833D[C4030300]00     <1>       cmp    word [u.pcount], 0
17379 00011D1B 7705                   <1>       ja     short dskw_4
17380                                  <1> dskw_3:
17381                                  <1>       ; [u.base] = virtual address to transfer (as source address)
17382 00011D1D E821FAFFFF             <1>       call   trans_addr_r ; translate virtual address to physical (r)
17383                                  <1> dskw_4:
17384 00011D22 BB[94070300]           <1>       mov    ebx, writei_buffer
17385                                  <1>       ; EBX (r5) = system (I/O) buffer address
17386 00011D27 E883FAFFFF             <1>       call   sioreg
17387                                  <1>       ; ESI = file (user data) offset
17388                                  <1>       ; EDI = sector (I/O) buffer offset
17389                                  <1>       ; ECX = byte count
17390                                  <1>       ;
17391 00011D2C F3A4                   <1>       rep    movsb
17392                                  <1>       ; 25/07/2015
17393                                  <1>       ; eax = remain bytes in buffer
17394                                  <1>       ;     (check if remain bytes in the buffer > [u.pcount])
17395 00011D2E 09C0                   <1>       or     eax, eax
17396 00011D30 75EB                   <1>       jnz    short dskw_3 ; (page end before system buffer end!)
17397                                  <1>
17398                                  <1>       ; 23/10/2016
17399 00011D32 B101                   <1>       mov    cl, 1
17400 00011D34 5E                     <1>       pop    esi
17401 00011D35 A1[44950100]           <1>       mov    eax, [writei.sector]
17402                                  <1>       ; esi = logical dos drive description table address
```

```
17403                                  <1>       ; eax = sector number
17404                                  <1>       ; ebx = writei buffer address
17405                                  <1>       ; ecx = sector count
17406 00011D3A E82F0D0000              <1>       call   disk_write ; / yes, write the block
17407 00011D3F 7307                    <1>       jnc    short dskw_5
17408                                  <1>
17409 00011D41 B812000000              <1>       mov    eax, 18 ; drive not ready or WRITE ERROR !
17410 00011D46 EB99                    <1>       jmp    short dskw_err
17411                                  <1>
17412                                  <1> dskw_5:
17413                                  <1>       ; 26/10/2016
17414 00011D48 0FB61D[64950100]        <1>       movzx  ebx, byte [writei.ofn] ; open file number
17415 00011D4F C0E302                  <1>       shl    bl, 2 ; *4
17416 00011D52 8B83[34990100]          <1>       mov    eax, [ebx+OF_POINTER]
17417 00011D58 3B83[5C990100]          <1>       cmp    eax, [ebx+OF_SIZE]
17418 00011D5E 7606                    <1>       jna    short dskw_6
17419 00011D60 8983[5C990100]          <1>       mov    [ebx+OF_SIZE], eax
17420                                  <1> dskw_6:
17421                                  <1>       ;shr   bl, 2
17422 00011D66 833D[88030300]00        <1>       cmp   dword [u.count], 0 ; / any more data to write?
17423 00011D6D 760A                    <1>       jna    short dskw_7
17424 00011D6F A1[54950100]            <1>       mov    eax, [writei.fclust]
17425 00011D74 E931FFFFFF              <1>       jmp    dskw_0 ; / yes, branch
17426                                  <1> dskw_7:
17427                                  <1>       ; update last modif. date&time of the file
17428                                  <1>       ; (also updates file size as OF_SIZE)
17429 00011D79 E82E030000              <1>       call   update_file_lmdt
17430                                  <1>       ;mov   byte [setfmod], 0
17431                                  <1>
17432                                  <1>       ; 03/08/2013
17433 00011D7E C605[C6030300]00        <1>       mov    byte [u.kcall], 0
17434                                  <1>       ; 23/10/2016
17435                                  <1>       ;mov   eax, [writei.fclust]
17436 00011D85 C3                      <1>       retn
17437                                  <1>
17438                                  <1> mget_w:
17439                                  <1>       ; 02/11/2016
17440                                  <1>       ; 01/11/2016
17441                                  <1>       ; 23/10/2016, 31/10/2016
17442                                  <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
17443                                  <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
17444                                  <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
17445                                  <1>       ;
17446                                  <1>       ; Get existing or (allocate) a new disk block for file
17447                                  <1>       ;
17448                                  <1>       ; INPUTS ->
17449                                  <1>       ;    [u.fofp] = file offset pointer
17450                                  <1>       ;    [i.size] = file size
17451                                  <1>       ;    [u.count] = byte count
17452                                  <1>       ;    EAX = First cluster
17453                                  <1>       ;    [cdev] = Logical dos drive number
17454                                  <1>       ;    [writei.ofn] = File Number
17455                                  <1>       ;                (Open file index, 0 based)
17456                                  <1>       ;    ([u.off] = file offset)
17457                                  <1>       ; OUTPUTS ->
17458                                  <1>       ;    EAX = logical sector number
17459                                  <1>       ;    ESI = Logical Dos Drive Description Table address
17460                                  <1>       ;
17461                                  <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI, EBP
17462                                  <1>
17463 00011D86 8B35[74030300]          <1>       mov   esi, [u.fofp]
17464 00011D8C 8B2E                    <1>       mov    ebp, [esi] ; u.off (or EBX*4+OF_POINTER)
17465                                  <1>
17466 00011D8E 29C9                    <1>       sub    ecx, ecx
17467 00011D90 8A2D[46030300]          <1>       mov    ch, [cdev]
17468                                  <1>
17469 00011D96 BE00010900              <1>       mov    esi, Logical_DOSDisks
17470 00011D9B 01CE                    <1>       add    esi, ecx
17471                                  <1>
17472                                  <1>       ; 31/10/2016
17473 00011D9D 89C3                    <1>       mov    ebx, eax ; First Cluster or FDT address
17474                                  <1>
17475 00011D9F 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
17476 00011DA3 0F86DD010000            <1>       jna    mget_w_14 ; Singlix FS
17477                                  <1>
17478 00011DA9 0FB74611                <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
17479 00011DAD 0FB65613                <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
17480 00011DB1 8815[42950100]          <1>       mov    [writei.spc], dl  ; sectors per cluster
17481 00011DB7 F7E2                    <1>       mul    edx
17482                                  <1>       ; edx = 0
17483                                  <1>       ; eax = bytes per cluster (<= 65536)
17484                                  <1>
17485                                  <1>       ; 02/11/2016
17486 00011DB9 89C1                    <1>       mov    ecx, eax
17487 00011DBB 48                      <1>       dec    eax
17488 00011DBC 66A3[48950100]          <1>       mov    [writei.bpc], ax
17489                                  <1>
17490 00011DC2 89E8                    <1>       mov    eax, ebp
17491 00011DC4 0305[88030300]          <1>       add    eax, [u.count] ; next file position
17492 00011DCA 3B05[55040300]          <1>       cmp    eax, [i.size] ; <= file size ?
17493 00011DD0 0F86FC000000            <1>       jna    mget_w_4 ; no
17494                                  <1>
17495 00011DD6 F7F1                    <1>       div    ecx
17496 00011DD8 A3[50950100]            <1>       mov    [writei.c_index], eax ; cluster index
17497                                  <1>       ; edx = byte offset in cluster (<= 65535)
17498                                  <1>       ;mov   [writei.offset], dx
17499                                  <1>       ;shr   dx, 9 ; / 512
17500                                  <1>       ;mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17501                                  <1>
17502 00011DDD 29D2                    <1>       sub    edx, edx ; 01/11/2016
17503 00011DDF 8915[44950100]          <1>       mov    [writei.sector], edx ; 0
17504 00011DE5 668915[4A950100]        <1>       mov    [writei.offset], dx  ; byte offset in cluster
17505 00011DEC 8815[43950100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17506                                  <1>
17507 00011DF2 89D8                    <1>       mov    eax, ebx ; First Cluster
```

```
17508                                  <1>
17509                                  <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
17510 00011DF4 3815[40950100]          <1>       cmp   byte [writei.valid], dl ; 0
17511 00011DFA 7624                    <1>       jna   short mget_w_0
17512                                  <1>
17513 00011DFC 8815[40950100]          <1>       mov   byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
17514                                  <1>
17515 00011E02 3B05[54950100]          <1>       cmp   eax, [writei.fclust]
17516 00011E08 7516                    <1>       jne   short mget_w_0
17517                                  <1>
17518 00011E0A 8A0D[46030300]          <1>       mov   cl, [cdev]
17519 00011E10 3A0D[41950100]          <1>       cmp   cl, [writei.drv]
17520 00011E16 7508                    <1>       jne   short mget_w_0
17521                                  <1>       ; [writei.l_clust] & [writei.l_index] are valid,
17522                                  <1>       ;  we don't need to get last cluster & last cluster index
17523 00011E18 8B0D[60950100]          <1>       mov   ecx, [writei.l_index]
17524 00011E1E EB64                    <1>       jmp   short mget_w_2
17525                                  <1> mget_w_0:
17526 00011E20 A3[54950100]            <1>       mov   [writei.fclust], eax ; first cluster
17527                                  <1>       ; edx = 0
17528 00011E25 A3[4C950100]            <1>       mov   [writei.cluster], eax ; first cluster ; 01/11/2016
17529 00011E2A 8915[58950100]          <1>       mov   [writei.fs_index], edx ; 0 ; curret cluster index
17530                                  <1>
17531                                  <1>       ; FAT file system (FAT12, FAT16, FAT32)
17532 00011E30 E8FCB5FFFF              <1>       call  get_last_cluster
17533 00011E35 0F822B010000            <1>       jc    mget_w_err ; eax = error code
17534                                  <1>
17535 00011E3B A3[5C950100]            <1>       mov   [writei.lclust], eax ; last cluster
17536                                  <1>
17537 00011E40 8B0D[80930100]          <1>       mov   ecx, [glc_index] ; last cluster index
17538 00011E46 890D[60950100]          <1>       mov   [writei.l_index], ecx
17539                                  <1>
17540 00011E4C A0[64950100]            <1>       mov   al, [writei.ofn]
17541 00011E51 FEC0                    <1>       inc   al
17542 00011E53 A2[9F950100]            <1>       mov   [setfmod], al ; update lm date&time sign
17543                                  <1>
17544                                  <1> mget_w_1:
17545 00011E58 3B0D[50950100]          <1>       cmp   ecx, [writei.c_index]  ; last cluster index
17546 00011E5E 7324                    <1>       jnb   short mget_w_2 ; 01/11/2016
17547                                  <1>
17548 00011E60 A1[5C950100]            <1>       mov   eax, [writei.lclust]
17549                                  <1>       ; EAX = Last cluster
17550 00011E65 E8D5B6FFFF              <1>       call  add_new_cluster
17551 00011E6A 0F82F6000000            <1>       jc    mget_w_err ; eax = error code
17552                                  <1>       ; edx = 0
17553 00011E70 A3[5C950100]            <1>       mov   [writei.lclust], eax ; (new) last cluster
17554 00011E75 8B0D[60950100]          <1>       mov   ecx, [writei.l_index]
17555 00011E7B 41                      <1>       inc   ecx ; add 1 to last cluster index
17556 00011E7C 890D[60950100]          <1>       mov   [writei.l_index], ecx ; current last cluster index
17557                                  <1>
17558 00011E82 EBD4                    <1>       jmp   short mget_w_1
17559                                  <1>
17560                                  <1> mget_w_2:
17561 00011E84 89E9                    <1>       mov   ecx, ebp
17562 00011E86 030D[88030300]          <1>       add   ecx, [u.count]
17563 00011E8C 890D[55040300]          <1>       mov   [i.size], ecx ; save new file size
17564                                  <1>       ;sub   edx, edx ; 0
17565                                  <1>
17566 00011E92 A0[46030300]            <1>       mov   al, [cdev]
17567 00011E97 A2[41950100]            <1>       mov   [writei.drv], al ; physical drive number
17568                                  <1>       ; edx = 0
17569 00011E9C 89E8                    <1>       mov   eax, ebp ; file offset
17570 00011E9E 0FB70D[48950100]        <1>       movzx ecx, word [writei.bpc] ; bytes per cluster - 1
17571 00011EA5 41                      <1>       inc   ecx ; bytes per cluster
17572 00011EA6 F7F1                    <1>       div   ecx
17573                                  <1>       ; edx = byte offset in cluster (<= 65535)
17574                                  <1>       ; eax = cluster index
17575 00011EA8 A3[50950100]            <1>       mov   [writei.c_index], eax
17576 00011EAD 668915[4A950100]        <1>       mov   [writei.offset], dx
17577 00011EB4 66C1EA09                <1>       shr   dx, 9 ; / 512
17578 00011EB8 8815[43950100]          <1>       mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17579                                  <1>
17580                                  <1> mget_w_3:
17581 00011EBE 3B05[60950100]          <1>       cmp   eax, [writei.l_index] ; last cluster index
17582 00011EC4 752A                    <1>       jne   short mget_w_5
17583                                  <1>
17584 00011EC6 A3[58950100]            <1>       mov   [writei.fs_index], eax ; cluster index (for next check)
17585 00011ECB A1[5C950100]            <1>       mov   eax, [writei.lclust] ; last cluster
17586 00011ED0 EB60                    <1>       jmp   short mget_w_10
17587                                  <1>
17588                                  <1> mget_w_4: ; 02/11/2016
17589                                  <1>       ; eax = next file position
17590 00011ED2 2B05[88030300]          <1>       sub   eax, [u.count] ; current file position
17591                                  <1>       ; edx = 0
17592                                  <1>       ; ecx = bytes per cluster
17593 00011ED8 F7F1                    <1>       div   ecx
17594 00011EDA A3[50950100]            <1>       mov   [writei.c_index], eax ; cluster index
17595 00011EDF 668915[4A950100]        <1>       mov   [writei.offset], dx
17596 00011EE6 66C1EA09                <1>       shr   dx, 9 ; / 512
17597 00011EEA 8815[43950100]          <1>       mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17598                                  <1>
17599                                  <1> mget_w_5:
17600 00011EF0 21C0                    <1>       and   eax, eax ; 0 = First Cluster's index number
17601 00011EF2 750C                    <1>       jnz   short mget_w_6
17602                                  <1>
17603 00011EF4 A3[58950100]            <1>       mov   [writei.fs_index], eax ; cluster index (for next check)
17604 00011EF9 A1[54950100]            <1>       mov   eax, [writei.fclust] ; first cluster
17605 00011EFE EB32                    <1>       jmp   short mget_w_10
17606                                  <1>
17607                                  <1> mget_w_6:
17608 00011F00 3B05[58950100]          <1>       cmp   eax, [writei.fs_index] ; current cluster index (>0)
17609 00011F06 7507                    <1>       jne   short mget_w_7
17610 00011F08 A1[4C950100]            <1>       mov   eax, [writei.cluster] ; current cluster
17611 00011F0D EB3A                    <1>       jmp   short mget_w_11
17612                                  <1>
```

```
17613                                  <1> mget_w_7:
17614 00011F0F 89C1                    <1>         mov    ecx, eax
17615 00011F11 2B0D[58950100]          <1>         sub    ecx, [writei.fs_index]
17616 00011F17 730D                    <1>         jnc    short mget_w_8
17617                                  <1>         ; get cluster by index from the first cluster
17618 00011F19 A1[54950100]            <1>         mov    eax, [writei.fclust]
17619 00011F1E 8B0D[50950100]          <1>         mov    ecx, [writei.c_index]
17620 00011F24 EB05                    <1>         jmp    short mget_w_9
17621                                  <1>
17622                                  <1> mget_w_8:
17623 00011F26 A1[4C950100]            <1>         mov    eax, [writei.cluster] ; beginning cluster
17624                                  <1>         ; ecx = cluster sequence number after the beginning cluster
17625                                  <1>         ; sub  edx, edx ; 0
17626                                  <1>
17627                                  <1> mget_w_9:
17628                                  <1>         ; EAX = Beginning cluster
17629                                  <1>         ; EDX = Sector index in disk/file section
17630                                  <1>         ;       (Only for SINGLIX file system!)
17631                                  <1>         ; ECX = Cluster sequence number after the beginning cluster
17632                                  <1>         ; ESI = Logical DOS Drive Description Table address
17633 00011F2B E815B7FFFF              <1>         call   get_cluster_by_index
17634 00011F30 7234                    <1>         jc     short mget_w_err ; error code in EAX
17635                                  <1>         ; EAX = Cluster number
17636                                  <1> mget_w_10:
17637 00011F32 A3[4C950100]            <1>         mov    [writei.cluster], eax ; FDT number for Singlix File System
17638                                  <1>
17639 00011F37 807E0300                <1>         cmp    byte [esi+LD_FATType], 0
17640 00011F3B 7638                    <1>         jna    short mget_w_13
17641                                  <1>         ; 01/11/2016
17642 00011F3D 8B15[50950100]          <1>         mov    edx, [writei.c_index]
17643 00011F43 8915[58950100]          <1>         mov    [writei.fs_index], edx
17644                                  <1> mget_w_11:
17645 00011F49 83E802                  <1>         sub    eax, 2
17646 00011F4C 0FB615[42950100]        <1>         movzx  edx, byte [writei.spc]
17647 00011F53 F7E2                    <1>         mul    edx
17648                                  <1>
17649 00011F55 034668                  <1>         add    eax, [esi+LD_DATABegin]
17650 00011F58 8A15[43950100]          <1>         mov    dl, [writei.s_index]
17651 00011F5E 01D0                    <1>         add    eax, edx
17652                                  <1> mget_w_12:
17653 00011F60 A3[44950100]            <1>         mov    [writei.sector], eax
17654                                  <1>         ;; buffer validation must be done in writei
17655                                  <1>         ;;mov  byte [writei.valid], 1
17656 00011F65 C3                      <1>         retn
17657                                  <1>
17658                                  <1> mget_w_err:
17659 00011F66 A3[C8030300]            <1>         mov    [u.error], eax
17660 00011F6B A3[64030300]            <1>         mov    [u.r0], eax
17661 00011F70 E937B8FFFF              <1>         jmp    error
17662                                  <1>
17663                                  <1> mget_w_13:
17664                                  <1>         ; EAX = FDT number (Current Section)
17665                                  <1>         ; EDX = Sector index from the first section (0,1,2,3,4...)
17666 00011F75 2B15[58950100]          <1>         sub    edx, [writei.fs_index]
17667                                  <1>         ; EDX = Sector index from current section
17668 00011F7B 8915[58950100]          <1>         mov    [writei.fs_index], edx
17669 00011F81 40                      <1>         inc    eax ; the first data sector in FS disk section
17670 00011F82 01D0                    <1>         add    eax, edx
17671 00011F84 EBDA                    <1>         jmp    short mget_w_12
17672                                  <1>
17673                                  <1> mget_w_14:
17674 00011F86 8A4E12                  <1>         mov    cl, [esi+LD_FS_BytesPerSec+1]
17675 00011F89 D0E9                    <1>         shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
17676 00011F8B 880D[42950100]          <1>         mov    [writei.spc], cl  ; sectors per cluster
17677                                  <1>         ; NOTE: writei bytes per sector value is always 512 !
17678 00011F91 66C705[48950100]00-     <1>         mov    word [writei.bpc], 512
17678 00011F99 02                      <1>
17679                                  <1>
17680 00011F9A 89E9                    <1>         mov    ecx, ebp
17681 00011F9C 030D[88030300]          <1>         add    ecx, [u.count] ; next file position
17682 00011FA2 3B0D[55040300]          <1>         cmp    ecx, [i.size] ; <= file size ?
17683 00011FA8 0F86C8000000            <1>         jna    mget_w_19 ; no
17684                                  <1>
17685 00011FAE 29D2                    <1>         sub    edx, edx ; 0
17686 00011FB0 8915[44950100]          <1>         mov    [writei.sector], edx ; 0
17687 00011FB6 668915[4A950100]        <1>         mov    [writei.offset], dx  ; byte offset in cluster
17688 00011FBD 8815[43950100]          <1>         mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17689                                  <1>
17690 00011FC3 C1E909                  <1>         shr    ecx, 9 ; 1 cluster = 512 bytes
17691 00011FC6 890D[50950100]          <1>         mov    [writei.c_index], ecx ; section/cluster index
17692                                  <1>
17693 00011FCC 89D8                    <1>         mov    eax, ebx ; FDT number (First FDT address)
17694                                  <1>
17695                                  <1>         ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
17696 00011FCE 3815[40950100]          <1>         cmp    byte [writei.valid], dl ; 0
17697 00011FD4 7624                    <1>         jna    short mget_w_15
17698                                  <1>
17699 00011FD6 8815[40950100]          <1>         mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
17700                                  <1>
17701 00011FDC 3B05[54950100]          <1>         cmp    eax, [writei.fclust]
17702 00011FE2 7516                    <1>         jne    short mget_w_15
17703                                  <1>
17704 00011FE4 8A0D[46030300]          <1>         mov    cl, [cdev]
17705 00011FEA 3A0D[41950100]          <1>         cmp    cl, [writei.drv]
17706 00011FF0 7508                    <1>         jne    short mget_w_15
17707                                  <1>         ; [writei.l_clust] & [writei.l_index] are valid,
17708                                  <1>         ;  we don't need to get last cluster & last cluster index
17709 00011FF2 8B0D[60950100]          <1>         mov    ecx, [writei.l_index]
17710 00011FF8 EB49                    <1>         jmp    short mget_w_17
17711                                  <1> mget_w_15:
17712 00011FFA A3[54950100]            <1>         mov    [writei.fclust], eax ; first section (FDT number)
17713                                  <1>         ; edx = 0
17714 00011FFF 8915[4C950100]          <1>         mov    [writei.cluster], edx ; 0 ; current section
17715 00012005 8915[58950100]          <1>         mov    [writei.fs_index], edx ; 0 ; curret section index
17716                                  <1>
```

```
                                <1>       ; eax = FDT number (section 0 header address)
0001200B E85FB6FFFF             <1>       call   get_last_section
00012010 0F8250FFFFFF           <1>       jc     mget_w_err ; eax = error code
                                <1>
00012016 8915[58950100]         <1>       mov    [writei.fs_index], edx ; sector index in last section
                                <1>
0001201C A3[5C950100]           <1>       mov    [writei.lclust], eax ; last section address
                                <1>
00012021 8B0D[80930100]         <1>       mov    ecx, [glc_index] ; last section index
00012027 890D[60950100]         <1>       mov    [writei.l_index], ecx
                                <1>
0001202D A0[64950100]           <1>       mov    al, [writei.ofn]
00012032 FEC0                   <1>       inc    al
00012034 A2[9F950100]           <1>       mov    [setfmod], al ; update lm date&time sign
                                <1>
                                <1> mget_w_16:
                                <1>       ; edx = (existing) last section (sector) index
00012039 8B0D[50950100]         <1>       mov    ecx, [writei.c_index] ; final section (sector) index
0001203F 29D1                   <1>       sub    ecx, edx
00012041 7633                   <1>       jna    short mget_w_19
                                <1>        ; ecx = sector count
                                <1> mget_w_17:
00012043 A1[5C950100]           <1>       mov    eax, [writei.lclust]
                                <1>       ; ESI = Logical dos drv desc. table address
                                <1>       ; EAX = Last section
                                <1>       ; (ECX = 0 for directory)
                                <1>       ; ECX = sector count (except FDT)
00012048 E8E5ABFFFF             <1>       call   add_new_fs_section
0001204D 7312                   <1>       jnc    short mget_w_18
                                <1>
                                <1>       ; If error number = 27h (insufficient disk space)
                                <1>       ; it is needed to check free consequent sectors
                                <1>       ; (1 data sector at least and +1 section header sector)
                                <1>
0001204F 83F827                 <1>       cmp    eax, 27h
00012052 0F850EFFFFFF           <1>       jne    mget_w_err ; eax = error code
                                <1>
                                <1>       ; ecx = count of free consequent sectors
                                <1>       ; ecx must be > 1 (1 data + 1 header sector)
00012058 49                     <1>       dec    ecx
00012059 0F8407FFFFFF           <1>       jz     mget_w_err
0001205F EBE2                   <1>       jmp    short mget_w_17
                                <1>
                                <1> mget_w_18:
00012061 A3[5C950100]           <1>       mov    [writei.lclust], eax ; (new) last section
                                <1>       ; ecx = sector count (except section header)
00012066 8B15[60950100]         <1>       mov    edx, [writei.l_index]
0001206C 01CA                   <1>       add    edx, ecx ; add sector count to index
0001206E 8915[60950100]         <1>       mov    [writei.l_index], edx
00012074 EBC3                   <1>       jmp    short mget_w_16
                                <1>
                                <1> mget_w_19:
00012076 89E9                   <1>       mov    ecx, ebp
00012078 030D[88030300]         <1>       add    ecx, [u.count]
0001207E 890D[55040300]         <1>       mov    [i.size], ecx ; save new file size
                                <1>       ;sub   edx, edx ; 0
                                <1>
00012084 A0[46030300]           <1>       mov    al, [cdev]
00012089 A2[41950100]           <1>       mov    [writei.drv], al ; physical drive number
                                <1>       ; edx = 0
0001208E 89E8                   <1>       mov    eax, ebp ; file offset
00012090 89C2                   <1>       mov    edx, eax
                                <1>       ; 1 cluster = 512 bytes (for Singlix FS)
00012092 C1E809                 <1>       shr    eax, 9  ; / 512
00012095 81E2FF010000           <1>       and    edx, 1FFh
                                <1>       ; edx = byte offset in cluster/sector (<= 511)
                                <1>       ; eax = section (sector/cluster) index
0001209B A3[50950100]           <1>       mov    [writei.c_index], eax
000120A0 668915[4A950100]       <1>       mov    [writei.offset], dx
                                <1>       ;mov   byte [writei.s_index], 0 ; sector index in cluster
000120A7 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>
000120AC 31C0                   <1>       xor    eax, eax
                                <1>
000120AE 0FB61D[9F950100]       <1>       movzx  ebx, byte [setfmod]
000120B5 FECB                   <1>       dec    bl ; open file index number (0 based)
                                <1>
000120B7 8AA3[0C990100]         <1>       mov    ah, [ebx+OF_DRIVE]
000120BD BE00010900             <1>       mov    esi, Logical_DOSDisks
000120C2 01C6                   <1>       add    esi, eax
000120C4 C0E302                 <1>       shl    bl, 2 ; *4
000120C7 8B8B[E4980100]         <1>       mov    ecx, [ebx+OF_FCLUSTER] ; first cluster
000120CD 8B93[AC990100]         <1>       mov    edx, [ebx+OF_DIRCLUSTER] ; dir cluster
```

```
17822                                  <1>
17823 000120D3 D0EB                    <1>       shr    bl, 1 ; /2
17824 000120D5 0FB7BB[4C9A0100]        <1>       movzx  edi, word [ebx+OF_DIRENTRY]
17825                                  <1>
17826 000120DC 803D[DC900100]01        <1>       cmp    byte [DirBuff_ValidData], 1
17827 000120E3 726E                    <1>       jb     short uflmdt_4
17828                                  <1>
17829 000120E5 A0[DA900100]            <1>       mov    al, [DirBuff_DRV]
17830 000120EA 2C41                    <1>       sub    al, 'A'
17831 000120EC 38E0                    <1>       cmp    al, ah
17832 000120EE 7563                    <1>       jne    short uflmdt_4 ; different drive
17833 000120F0 8A4603                  <1>       mov    al, [esi+LD_FATType]
17834 000120F3 3A05[DB900100]          <1>       cmp    al, [DirBuff_FATType]
17835 000120F9 755B                    <1>       jne    short uflmdt_5 ; different FS type
17836 000120FB 3B15[E1900100]          <1>       cmp    edx, [DirBuff_Cluster]
17837 00012101 7553                    <1>       jne    short uflmdt_5 ; different cluster
17838                                  <1>
17839                                  <1> uflmdt_1:
17840                                  <1>       ; Directory buffer is ready here!
17841                                  <1>       ; OF_FCLUSTER must be compared/verified
17842 00012103 BE00000800              <1>       mov    esi, Directory_Buffer
17843 00012108 66C1E705                <1>       shl    di, 5 ; dir entry index * 32
17844 0001210C 01FE                    <1>       add    esi, edi ; offset
17845                                  <1>       ;
17846 0001210E F6460B18                <1>       test   byte [esi+DirEntry_Attr], 18h ; Vol & Dir
17847 00012112 750F                    <1>       jnz    short uflmdt_2 ; not a valid file !
17848 00012114 668B4614                <1>       mov    ax, [esi+DirEntry_FstClusHI]
17849 00012118 C1E010                  <1>       shl    eax, 16
17850 0001211B 668B461A                <1>       mov    ax, [esi+DirEntry_FstClusLO]
17851 0001211F 39C8                    <1>       cmp    eax, ecx ; same first cluster ?
17852 00012121 7407                    <1>       je     short uflmdt_3 ; yes, it is OK !!!
17853                                  <1>
17854                                  <1> uflmdt_2:
17855                                  <1>       ; save directory buffer if has modified/changed sign
17856                                  <1>       ; (It is good to save dir buff even if the searched
17857                                  <1>       ; directory entry is not found !?)
17858 00012123 E85E98FFFF              <1>       call   save_directory_buffer
17859 00012128 F9                      <1>       stc    ; update failed
17860 00012129 C3                      <1>       retn
17861                                  <1>
17862                                  <1> uflmdt_3:
17863                                  <1>       ; Update directory entry
17864                                  <1>       ; 26/10/2016
17865 0001212A D0E3                    <1>       shl    bl, 1 ; *2
17866 0001212C 8B83[5C990100]          <1>       mov    eax, [ebx+OF_SIZE] ; file size
17867 00012132 89461C                  <1>       mov    [esi+DirEntry_FileSize], eax
17868                                  <1>       ;
17869 00012135 E8AE97FFFF              <1>       call   convert_current_date_time
17870                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
17871                                  <1>         ;       AX = Time in dos dir entry format
17872 0001213A 66894616                <1>       mov    [esi+DirEntry_WrtTime], ax
17873 0001213E 66895618                <1>       mov    [esi+DirEntry_WrtDate], dx
17874 00012142 66895612                <1>       mov    [esi+DirEntry_LastAccDate], dx
17875 00012146 C605[DC900100]02        <1>       mov    byte [DirBuff_ValidData], 2
17876 0001214D E83498FFFF              <1>       call   save_directory_buffer
17877 00012152 C3                      <1>       retn
17878                                  <1>
17879                                  <1> uflmdt_4:
17880                                  <1>       ; Directory buffer sector read&write
17881                                  <1>       ; 23/10/2016
17882                                  <1>       ;
17883 00012153 8A4603                  <1>       mov    al, [esi+LD_FATType]
17884                                  <1> uflmdt_5:
17885 00012156 BB[9C090300]            <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
17886                                  <1>
17887 0001215B 20C0                    <1>       and    al, al ; 0 = Singlix FS
17888 0001215D 0F8492000000            <1>       jz     uflmdt_11
17889                                  <1>
17890 00012163 21D2                    <1>       and    edx, edx
17891 00012165 7521                    <1>       jnz    short uflmdt_9
17892                                  <1>
17893 00012167 3C02                    <1>       cmp    al, 2  ; 3 = FAT32
17894 00012169 771A                    <1>       ja     short uflmdt_8
17895                                  <1>
17896 0001216B 89F8                    <1>       mov    eax, edi ; directory entry index number
17897 0001216D 66C1E804                <1>       shr    ax, 4 ; 16 entries per sector
17898 00012171 034664                  <1>       add    eax, [esi+LD_ROOTBegin]
17899                                  <1>       ; eax = root directory sector
17900                                  <1> uflmdt_6:
17901 00012174 50                      <1>       push   eax ; * ; disk sector address
17902 00012175 51                      <1>       push   ecx ; first cluster
17903 00012176 B901000000              <1>       mov    ecx, 1
17904                                  <1>       ; ecx = sector count
17905 0001217B E8FD080000              <1>       call   disk_read
17906 00012180 59                      <1>       pop    ecx
17907 00012181 731A                    <1>       jnc    short uflmdt_10
17908 00012183 58                      <1>       pop    eax ; *
17909                                  <1> uflmdt_7:
17910 00012184 C3                      <1>       retn
17911                                  <1>
17912                                  <1> uflmdt_8:
17913 00012185 8B5632                  <1>       mov    edx, [esi+LD_BPB+FAT32_RootFClust]
17914                                  <1> uflmdt_9:
17915 00012188 83FA02                  <1>       cmp    edx, 2
17916 0001218B 72F7                    <1>       jb     short uflmdt_7 ; invalid, nothing to do
17917                                  <1>
17918 0001218D 83EA02                  <1>       sub    edx, 2
17919 00012190 89D0                    <1>       mov    eax, edx
17920 00012192 0FB65613                <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
17921 00012196 F7E2                    <1>       mul    edx
17922 00012198 034668                  <1>       add    eax, [esi+LD_DATABegin]
17923                                  <1>       ; eax = sub directory (data) sector
17924 0001219B EBD7                    <1>       jmp    short uflmdt_6
17925                                  <1>
17926                                  <1> uflmdt_10:
```

```
17927                                  <1>       ; Directory sector buffer is ready here!
17928                                  <1>       ; OF_FCLUSTER must be compared/verified
17929                                  <1>       ; edi = dir entry index number (<= 2047)
17930 0001219D 6683E70F                <1>       and    di, 0Fh ; 16 entries per sector
17931 000121A1 66C1E705                <1>       shl    di, 5 ; dir entry index * 32
17932 000121A5 81C7[9C090300]          <1>       add    edi, rw_buffer
17933                                  <1>       ;
17934 000121AB F6470B18                <1>       test   byte [edi+DirEntry_Attr], 18h ; Vol & Dir
17935 000121AF 0F856EFFFFFF            <1>       jnz    uflmdt_2 ; not a valid file !
17936 000121B5 668B5714                <1>       mov    dx, [edi+DirEntry_FstClusHI]
17937 000121B9 C1E210                  <1>       shl    edx, 16
17938 000121BC 668B571A                <1>       mov    dx, [edi+DirEntry_FstClusLO]
17939 000121C0 39CA                    <1>       cmp    edx, ecx ; same first cluster ?
17940 000121C2 0F855BFFFFFF            <1>       jne    uflmdt_2 ; no !?
17941                                  <1>
17942                                  <1>       ; Update directory entry
17943 000121C8 E81B97FFFF              <1>       call   convert_current_date_time
17944                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
17945                                  <1>         ;       AX = Time in dos dir entry format
17946 000121CD 66894716                <1>       mov    [edi+DirEntry_WrtTime], ax
17947 000121D1 66895718                <1>       mov    [edi+DirEntry_WrtDate], dx
17948 000121D5 66895712                <1>       mov    [edi+DirEntry_LastAccDate], dx
17949                                  <1>
17950 000121D9 58                      <1>       pop    eax ; *
17951                                  <1>
17952 000121DA BB[9C090300]            <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
17953 000121DF B901000000              <1>       mov    ecx, 1
17954                                  <1>       ; esi = logical dos description table address
17955                                  <1>       ; eax = disk sector number/address (LBA)
17956                                  <1>       ; ecx = sector count
17957                                  <1>       ; ebx = buffer address
17958 000121E4 E885080000              <1>       call   disk_write
17959 000121E9 0F8234FFFFFF            <1>       jc     uflmdt_2
17960                                  <1>
17961                                  <1>       ; save directory buffer if has modified/changed sign
17962 000121EF E89297FFFF              <1>       call   save_directory_buffer
17963 000121F4 C3                      <1>       retn
17964                                  <1>
17965                                  <1> uflmdt_11:
17966                                  <1>       ; 24/10/2016
17967                                  <1>       ; Update last modification date & time of a file
17968                                  <1>       ; on a disk with Singlix File System.
17969                                  <1>       ;
17970                                  <1>       ; (Method: Read the FDT -File Description Table-
17971                                  <1>       ; sector of the file and update the lmdt data fields,
17972                                  <1>       ; then write FDT sector to the disk.
17973                                  <1>       ; /// It is easy but there is compatibility buffer
17974                                  <1>       ; method also for changing directory entry data and
17975                                  <1>       ; also there are some programming issues for Singlix
17976                                  <1>       ; file system (TRFS), which are not completed yet!)
17977                                  <1>       ;
17978                                  <1>       ; Not ready yet ! (24/10/2016)
17979                                  <1>       ; /// Temporary code for error return ! ///
17980 000121F5 31C0                    <1>       xor    eax, eax
17981 000121F7 F9                      <1>       stc
17982 000121F8 C3                      <1>       retn
17983                                  <1>
17984                                  <1> sysalloc:
17985                                  <1>       ; 14/10/2017
17986                                  <1>       ; 20/08/2017, 01/09/2017
17987                                  <1>       ; 20/02/2017, 04/03/2017, 15/05/2017
17988                                  <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
17989                                  <1>       ; (TRDOS 386 feature only!)
17990                                  <1>       ;
17991                                  <1>       ; Allocate Contiguous Memory Block/Pages (for user)
17992                                  <1>       ; (System call for DMA Buffer allocation etc.)
17993                                  <1>       ;
17994                                  <1>       ; INPUT ->
17995                                  <1>       ;     EBX = Virtual address (for user)
17996                                  <1>       ;          (Physical memory block/aperture
17997                                  <1>       ;          will be mapped to this virtual address)
17998                                  <1>       ;     ECX = Byte Count
17999                                  <1>       ;          (will be rounded up to page border)
18000                                  <1>       ;     If ECX = 0
18001                                  <1>       ;         System call will return with an error (cf=1)
18002                                  <1>       ;         but ECX will contain maximum size of
18003                                  <1>       ;         available memory aperture and physical
18004                                  <1>       ;         (beginning) address of that aperture
18005                                  <1>       ;         (which have maximum size) will be in EAX.
18006                                  <1>       ;     EDX = Upper limit of the requested physical memory
18007                                  <1>       ;          block/pages.
18008                                  <1>       ;         (The last byte address of the memory aperture
18009                                  <1>       ;          must not be equal to or above this limit.)
18010                                  <1>       ;     If EDX = 0
18011                                  <1>       ;        there is NOLIMIT !
18012                                  <1>       ;     If EDX = 0FFFFFFFFh (-1)
18013                                  <1>       ;        ESI = Lower Limit !
18014                                  <1>       ;            (Beginning of the block must not be 'less'
18015                                  <1>       ;            than this.) (Must be equal to or above...)
18016                                  <1>       ;        EDI = Upper Limit !
18017                                  <1>       ;            (End of the block must be !less! than this)
18018                                  <1>       ;            (The last byte addr of the memory aperture
18019                                  <1>       ;            must not be equal to or above this limit.)
18020                                  <1>       ;
18021                                  <1>       ; OUTPUT ->
18022                                  <1>       ;     If CF = 0
18023                                  <1>       ;     EAX = Physical address of the allocated memory block
18024                                  <1>       ;     ECX = Allocated bytes (as rounded up to page borders)
18025                                  <1>       ;     EBX = Virtual address (as rounded up)
18026                                  <1>       ;     IF CF = 1
18027                                  <1>       ;         Requested (size of) Memory block could not be
18028                                  <1>       ;         allocated to the user!
18029                                  <1>       ;     IF CF = 1 & EAX = 0 (Insufficient memory error!)
18030                                  <1>       ;        ECX = Total number of free bytes
18031                                  <1>       ;              (not size of available contiguous bytes!)
```

```
18032                                  <1>       ;      If CF = 1 & EAX > 0
18033                                  <1>       ;         there is not a memory aperture with requested size
18034                                  <1>       ;         but total free mem is not less than requested size.
18035                                  <1>       ;         EAX = Physical addr of available memory aperture
18036                                  <1>       ;              with max size
18037                                  <1>       ;               (but it doesn't fit to the conditions!)
18038                                  <1>       ;         ECX = Size of available memory aperture in bytes.
18039                                  <1>       ;      If CF = 1 -> EAX = 0FFFFFFFFh
18040                                  <1>       ;         Conditions/Parameters are wrong !
18041                                  <1>       ;         ECX is same with input value.
18042                                  <1>       ;
18043                                  <1>       ; Note:      Previously allocated pages will be deallocated if
18044                                  <1>       ;       new allocation conditions are met.
18045                                  <1>       ;
18046                                  <1>       ; Note: u.break control may be included in future versions
18047                                  <1>       ;
18048                                  <1>
18049 000121F9 31C0                   <1>       xor    eax, eax ; 0
18050                                  <1>       ; 14/10/2017
18051 000121FB 4A                     <1>       dec    edx ; is there a limit ?
18052 000121FC 7810                   <1>       js     short sysalloc_1 ;  0 -> 0FFFFFFFFh -> NO LIMIT
18053 000121FE 42                     <1>       inc    edx ; > 0
18054                                  <1>       ; Check upper address limit
18055                                  <1>       ;(round up to page borders)
18056 000121FF 81C1FF0F0000           <1>       add    ecx, PAGE_SIZE-1 ; 4095
18057 00012205 6681E100F0             <1>       and    cx, ~PAGE_OFF ; not 4095
18058 0001220A 39CA                   <1>       cmp    edx, ecx ; upper limit - block size
18059 0001220C 7224                   <1>       jb     short sysalloc_err
18060                                  <1> sysalloc_1:
18061                                  <1>       ; EAX = Beginning address (physical)
18062                                  <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
18063                                  <1>       ; ECX = Number of bytes to be allocated
18064 0001220E E83E41FFFF             <1>       call   allocate_memory_block
18065 00012213 721D                   <1>       jc     short sysalloc_err
18066                                  <1>       ; 01/09/2017
18067 00012215 29C2                   <1>       sub    edx, eax ; upper limit address - beginning address
18068 00012217 760F                   <1>       jna    short sysalloc_3 ; begin addr not less than the limit
18069 00012219 39CA                   <1>       cmp    edx, ecx
18070 0001221B 720B                   <1>       jb     short sysalloc_3 ; end address overs the limit
18071                                  <1> sysalloc_2:
18072                                  <1>       ; EAX = Beginning (physical) addr of the allocated mem block
18073                                  <1>       ; ECX = Num of allocated bytes (rounded up to page borders)
18074 0001221D 50                     <1>       push   eax ; * ; 04/03/2017
18075                                  <1>       ; Here, requested contiquous memory pages have been allocated
18076                                  <1>       ; on Memory Allocation Table but user's page directory
18077                                  <1>       ; and page tables have not been updated yet!
18078 0001221E 51                     <1>       push   ecx ; **
18079                                  <1>       ; ebx = virtual address (will be rounded up to page border)
18080                                  <1>       ; ecx = number of bytes to be deallocated
18081                                  <1>       ;      will be adjusted to ebx+ecx round down - ebx round up
18082 0001221F E88844FFFF             <1>       call   deallocate_user_pages
18083 00012224 731F                   <1>       jnc    short sysalloc_4 ; EAX = Deallocated memory bytes
18084 00012226 59                     <1>       pop    ecx ; **
18085 00012227 58                     <1>       pop    eax ; *
18086                                  <1> sysalloc_3:
18087                                  <1>       ; error !
18088                                  <1>       ; restore Memory Allocation Table Content
18089 00012228 E83143FFFF             <1>       call   deallocate_memory_block
18090 0001222D 31C0                   <1>       xor    eax, eax ; 0
18091 0001222F 48                     <1>       dec    eax ; 0FFFFFFFFh ; 15/05/2017
18092 00012230 EB09                   <1>       jmp    short sysalloc_wrong
18093                                  <1> sysalloc_err:
18094 00012232 8B2D[60030300]         <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
18095 00012238 894D18                 <1>       mov    [ebp+24], ecx ; return to user with ecx value
18096                                  <1> sysalloc_wrong:
18097                                  <1>       ; eax = 0FFFFFFFFh
18098 0001223B A3[64030300]           <1>       mov    [u.r0], eax
18099 00012240 E967B5FFFF             <1>       jmp    error
18100                                  <1> sysalloc_4:
18101 00012245 8B2D[60030300]         <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
18102 0001224B 894518                 <1>       mov    [ebp+24], eax ; return to user with ecx value
18103 0001224E 895D10                 <1>       mov    [ebp+16], ebx ; new value of ebx (rounded up)
18104 00012251 89C1                   <1>       mov    ecx, eax ; byte count (from 'deallocate_user_pages')
18105 00012253 5A                     <1>       pop    edx ; ** ; discard (another) byte count
18106 00012254 58                     <1>       pop    eax ; *
18107 00012255 A3[64030300]           <1>       mov    [u.r0], eax ; physical address
18108                                  <1>
18109 0001225A 51                     <1>       push   ecx ; 20/08/2017
18110                                  <1>       ;
18111                                  <1>       ; Write newly allocated contiguous (physical) pages
18112                                  <1>       ; on page dir and page tables of current user/process
18113                                  <1>       ; as PRESENT, USER, WRITABLE
18114                                  <1>       ; (then clear allocated pages)
18115 0001225B E84145FFFF             <1>       call   allocate_user_pages
18116                                  <1>       ;jnc   sysret ; OK! return to process with success...
18117                                  <1>
18118                                  <1>       ; 20/08/2017 ('sysdma' modification)
18119 00012260 59                     <1>       pop    ecx
18120 00012261 A1[64030300]           <1>       mov    eax, [u.r0]   ; physical address (of the block)
18121                                  <1>
18122 00012266 721D                   <1>       jc     short sysalloc_6
18123                                  <1>
18124 00012268 833D[B49F0100]FF       <1>       cmp    dword [dma_addr], 0FFFFFFFFh ; -1
18125 0001226F 0F8257B5FFFF           <1>       jb     sysret
18126                                  <1>
18127 00012275 A3[B49F0100]           <1>       mov    [dma_addr], eax ; save dma address for sysdma
18128 0001227A 890D[B89F0100]         <1>       mov    [dma_size], ecx ; save dma buff size for sysdma
18129                                  <1>
18130 00012280 E947B5FFFF             <1>       jmp    sysret
18131                                  <1>
18132                                  <1> sysalloc_6:
18133                                  <1>       ;
18134                                  <1>       ; unexpected error ! insufficient memory !? conflict !?
18135                                  <1>       ; (!!?there is not a free page for a new page table?!!)
18136                                  <1>       ; We need to terminate process with error message !!!
```

```
18137                                  <1>       ;
18138 00012285 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
18139 0001228B 8B4D18                  <1>       mov    ecx, [ebp+24] ; byte count
18140                                  <1>
18141                                  <1>       ; 20/08/2017
18142                                  <1>       ;mov   eax, [u.r0]   ; physical address (of the block)
18143                                  <1>
18144                                  <1>       ;
18145                                  <1>       ; restore Memory Allocation Table Content
18146 0001228E E8CB42FFFF              <1>       call   deallocate_memory_block
18147                                  <1>       ;
18148 00012293 803D[9A6E0000]03        <1>       cmp    byte [CRT_MODE], 3 ; 80x25 text mode?
18149 0001229A 7407                    <1>       je     short sysalloc_7 ; yes
18150                                  <1>       ; Current mode is VGA (or CGA graphics) mode,
18151                                  <1>       ; We need to return to text mode for displaying
18152                                  <1>       ; error message just before 'sysexit'.
18153 0001229C B003                    <1>       mov    al, 3
18154 0001229E E85FF8FEFF              <1>       call   _set_mode
18155                                  <1> sysalloc_7:
18156 000122A3 BE[E4420100]            <1>       mov    esi, beep_Insufficient_Memory ; error message
18157 000122A8 E87851FFFF              <1>       call   print_msg ; print/display the message
18158 000122AD B801000000              <1>       mov    eax, 1 ; ax=1 is needed for 'sysexit' procedure
18159 000122B2 E99CB6FFFF              <1>       jmp    sysexit ; and terminate the process !
18160                                  <1>
18161                                  <1> sysdalloc:
18162                                  <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
18163                                  <1>       ; (TRDOS 386 feature only!)
18164                                  <1>       ;
18165                                  <1>       ; Deallocate Memory Block/Pages (for user)
18166                                  <1>       ; (Complementary call for sysalloc.)
18167                                  <1>       ;
18168                                  <1>       ; INPUT ->
18169                                  <1>       ;      EBX = Virtual address (for user)
18170                                  <1>       ;            (will be rounded up to page border)
18171                                  <1>       ;      ECX = Byte Count
18172                                  <1>       ;           (will be adjusted to page borders)
18173                                  <1>       ;      If ICX = 0
18174                                  <1>       ;         nothing to do
18175                                  <1>       ;      If EBX + ECX > User's ESP
18176                                  <1>       ;         nothing to do
18177                                  <1>       ;
18178                                  <1>       ; Note: u.break control may be included in future versions
18179                                  <1>       ;
18180                                  <1>       ; OUTPUT ->
18181                                  <1>       ;      If CF = 0
18182                                  <1>       ;         EAX = Deallocated memory bytes
18183                                  <1>       ;         EBX = Virtual address (as rounded up)
18184                                  <1>       ;      IF CF = 1
18185                                  <1>       ;         EAX = 0
18186                                  <1>       ;
18187                                  <1>       ; Note:      Main purpose of this call is to deallocate/release
18188                                  <1>       ;      previously allocated (physically) contiguous memory
18189                                  <1>       ;      pages but beginning (virtual) address may not be
18190                                  <1>       ;      followed by physically contiguous pages. So, this
18191                                  <1>       ;      system call will deallocate user's virtually
18192                                  <1>       ;      contiguous memory pages. Also, there is not any
18193                                  <1>       ;      objections to use this system call without sysalloc
18194                                  <1>       ;      system call; only possible objection is to lost data
18195                                  <1>       ;      within user's memory space, if the beginning address
18196                                  <1>       ;      and size is not proper.
18197                                  <1>       ;
18198                                  <1>       ; Note: Empty page tables will not be deallocated!!!
18199                                  <1>       ;       (they will be deallocated at process termination)
18200                                  <1>       ;
18201                                  <1>       ; Note: When the program terminates itself or when it is
18202                                  <1>       ;      terminated by operating system kernel, all allocated
18203                                  <1>       ;      memory pages will be deallocated during termination
18204                                  <1>       ;      stage. So, 'sysdalloc' is not necessary except
18205                                  <1>       ;      forgiving memory block to other programs/processes.
18206                                  <1>       ;
18207 000122B7 8B15[5C030300]          <1>       mov    edx, [u.sp]
18208 000122BD 8B420C                  <1>       mov    eax, [edx+12] ; user's stack pointer
18209 000122C0 29C8                    <1>       sub    eax, ecx ; esp - byte count
18210 000122C2 24FC                    <1>       and    al, 0FCh ; dword alignment
18211 000122C4 39D8                    <1>       cmp    eax, ebx
18212 000122C6 7220                    <1>       jb     short sysdalloc_err ; deallocation overlaps with stack
18213                                  <1>
18214 000122C8 31C0                    <1>       xor    eax, eax
18215 000122CA 21C9                    <1>       and    ecx, ecx
18216 000122CC 7407                    <1>       jz     short sysdalloc_2
18217                                  <1>
18218 000122CE E8D943FFFF              <1>       call   deallocate_user_pages
18219 000122D3 7213                    <1>       jc     short sysdalloc_err
18220                                  <1>
18221                                  <1> sysdalloc_2:
18222 000122D5 A3[64030300]            <1>       mov    [u.r0], eax
18223 000122DA 8B2D[60030300]          <1>       mov    ebp, [u.usp]
18224 000122E0 895D10                  <1>       mov    [ebp+16], ebx ; new value of ebx
18225 000122E3 E9E4B4FFFF              <1>       jmp    sysret
18226                                  <1>
18227                                  <1> sysdalloc_err:
18228 000122E8 A3[64030300]            <1>       mov    [u.r0], eax ; 0
18229 000122ED E9BAB4FFFF              <1>       jmp    error
18230                                  <1>
18231                                  <1> syscalbac:
18232                                  <1>       ; SYS CALLBACK
18233                                  <1>       ; 03/08/2020
18234                                  <1>       ; 16/04/2017
18235                                  <1>       ; 14/04/2017
18236                                  <1>       ; 13/04/2017
18237                                  <1>       ; 28/02/2017
18238                                  <1>       ; 26/02/2017
18239                                  <1>       ; 24/02/2017
18240                                  <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
18241                                  <1>       ; (TRDOS 386 feature only!)
```

```
18242                                  <1>       ;
18243                                  <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
18244                                  <1>       ;
18245                                  <1>       ; INPUT ->
18246                                  <1>       ;     BL = IRQ number (Hardware interrupt request number)
18247                                  <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
18248                                  <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
18249                                  <1>       ;          (numbers >15 are invalid)
18250                                  <1>       ;
18251                                  <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
18252                                  <1>       ;          1 = Link IRQ by using Signal Response Byte method
18253                                  <1>       ;          2 = Link IRQ by using Callback service method
18254                                  <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
18255                                  <1>       ;         >3 = invalid
18256                                  <1>       ;
18257                                  <1>       ;     CL = Signal Return/Response Byte value
18258                                  <1>       ;
18259                                  <1>       ;     If BH = 3, kernel will put a counter value  ; 03/08/2020
18260                                  <1>       ;               (into the S.R.B. addr)
18261                                  <1>       ;             between 0 to 255. (start value = CL+1)
18262                                  <1>       ;
18263                                  <1>       ;     NOTE: counter value, for example: even and odd numbers
18264                                  <1>       ;           may be used for -audio- DMA buffer switch
18265                                  <1>       ;           within double buffer method, etc.
18266                                  <1>       ;
18267                                  <1>       ;     EDX = Signal return (Response) byte address
18268                                  <1>       ;                         - or -
18269                                  <1>       ;           Interrupt/Callback service/routine address
18270                                  <1>       ;
18271                                  <1>       ;           (virtual address in user's memory space)
18272                                  <1>       ;
18273                                  <1>       ; OUTPUT ->
18274                                  <1>       ;     CF = 0 & EAX = 0 -> Successful setting
18275                                  <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
18276                                  <1>       ;                  by another process
18277                                  <1>       ;            eax = ERR_PERM_DENIED -> prohibited or locked
18278                                  <1>       ;            eax = ERR_INV_PARAMETER ->
18279                                  <1>       ;                  invalid parameter/option or bad address
18280                                  <1>       ;
18281                                  <1>       ;     NOTE: Timer callbacks are set by using 'systimer'
18282                                  <1>       ;           system call (IRQ 0, PIT and IRQ 8, RTC)
18283                                  <1>       ;
18284                                  <1>       ;           Direct keyboard access is performed by using
18285                                  <1>       ;           Keyboard Interrupt (INT 32h)
18286                                  <1>       ;
18287                                  <1>       ;           It is prohibited here because:
18288                                  <1>       ;            1) Signal Response Byte method has not advantage
18289                                  <1>       ;               against INT 32h, function AH = 1. Also,
18290                                  <1>       ;               keyboard service interrupt will return with
18291                                  <1>       ;               ascii and scan codes (AL, AH) while
18292                                  <1>       ;               SRB method has only 1 byte space for ascii code
18293                                  <1>       ;               or scan code. One byte signal response is used
18294                                  <1>       ;               for ensuring very simple and very fast
18295                                  <1>       ;               virtual to physical memory address conversion
18296                                  <1>       ;               without any memory page crossover risk.
18297                                  <1>       ;               (Otherwise double page conversion or word
18298                                  <1>       ;               alignment would be needed.)
18299                                  <1>       ;            2) Badly written user code (callback code)
18300                                  <1>       ;               can prevent keyboard and timesharing functions
18301                                  <1>       ;               of the operating system via continuous and long
18302                                  <1>       ;               keyboard event handling by callback service.
18303                                  <1>       ;               (It can cause to lose immediate keystroke
18304                                  <1>       ;               response from hardware to user.)
18305                                  <1>       ;            3) If user will check any keyboard events, 'getkey'
18306                                  <1>       ;               (or 'getchar') must have more priority than other
18307                                  <1>       ;               (video etc.) events because only control ability
18308                                  <1>       ;               on a procedural infinite loop is a keyboard or
18309                                  <1>       ;               mouse event. So user can use keyboard function
18310                                  <1>       ;               at the end or at the beginning of a loop.
18311                                  <1>       ;               In this case, INT 32h is used for that purpose
18312                                  <1>       ;               and timer interrupt etc. callbacks can be used
18313                                  <1>       ;               for dynamic and synchronized data refresh/transfer
18314                                  <1>       ;               while cpu is in a static loop (without polling).
18315                                  <1>       ;               Keyboard Int callback is not more useful because
18316                                  <1>       ;               already a manual check (a key is pressed or not)
18317                                  <1>       ;               can be performed (via INT 32h, AH = 1) efficiently
18318                                  <1>       ;               in a loop to prevent a locked infinitive loop.
18319                                  <1>       ;
18320                                  <1>       ;            Disk IRQs (6,14,15) have been phohibited from ring 3
18321                                  <1>       ;            callback because, disk operations (file system services
18322                                  <1>       ;            etc.) are independent from user program, for fast disk r/w.
18323                                  <1>       ;            They are not more useful at ring 3 while they are in use
18324                                  <1>       ;            by standard diskio functions which are mandatory part of
18325                                  <1>       ;            (monolithic) OS kernel and mainprog command interpreter.
18326                                  <1>       ;            INT 33h diskio functions are enough for user level disk
18327                                  <1>       ;            r/w.
18328                                  <1>       ;
18329                                  <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
18330                                  <1>       ;
18331                                  <1>       ;           [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
18332                                  <1>       ;                         which IRQs are locked by (that) user.
18333                                  <1>       ;                           Lock and unlock (by user) will change
18334                                  <1>       ;                         these flags or 'terminate process' (sysexit)
18335                                  <1>       ;                         will clear these flags and unlock those IRQs.
18336                                  <1>       ;
18337                                  <1>       ;                         Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
18338                                  <1>       ;
18339                                  <1>       ;           IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
18340                                  <1>       ;
18341                                  <1>       ;           IRQ(x).method : 1 byte for callback method & status
18342                                  <1>       ;                           0 = Signal Response Byte method
18343                                  <1>       ;                           1 = Callback service method
18344                                  <1>       ;                           >1 = invalid for current 'syscalback'.
18345                                  <1>       ;                         or(+) 80h = IRQ is in use by system (ring 0)
18346                                  <1>       ;                                     function (audio etc.) or
```

```
18347                                  <1>       ;                          a device driver.
18348                                  <1>       ;                  (system function will ignore the lock/owner)
18349                                  <1>       ;
18350                                  <1>       ;        IRQ(x).srb: 1 byte, Signal Return/Response byte value
18351                                  <1>       ;                     (a fixed value by user or a counter value
18352                                  <1>       ;                    from 0 to 255, which is increased by every
18353                                  <1>       ;                    interrupt just before putting it into
18354                                  <1>       ;                    the Signal Response byte address
18355                                  <1>       ;                    (This is not used in callback serv method)
18356                                  <1>       ;
18357                                  <1>       ;        IRQ(x).addr      : 1 dword
18358                                  <1>       ;                    Signal Response Byte address (physical)
18359                                  <1>       ;                               -or-
18360                                  <1>       ;                    Callback service address (virtual)
18361                                  <1>       ;
18362                                  <1>       ;        IRQ(x).dev: 1 byte
18363                                  <1>       ;                    0 = Default device or kernel function
18364                                  <1>       ;                               -or-
18365                                  <1>       ;                    1-255 = Assigned device driver number
18366                                  <1>       ;
18367                                  <1>       ;        (x) = 3,4,5,7,9,10,11,12,13
18368                                  <1>       ;
18369                                  <1>       ;
18370                                  <1>       ;     NOTE: If user's process/program calls the kernel (INT 40h)
18371                                  <1>       ;           while it is already running in a (ring 3) callback
18372                                  <1>       ;           service, kernel will force (convert) system call to
18373                                  <1>       ;           'sysrele' (sys release). So, this feature provides
18374                                  <1>       ;           easy and simple usage of callback services without
18375                                  <1>       ;           falling into deepless <please 'callback me' then
18376                                  <1>       ;           let me 'callback you'> cycles! (User must return
18377                                  <1>       ;           from callback service by using 'sysrele' system
18378                                  <1>       ;           call, without a significant delay. Otherwise user
18379                                  <1>       ;           process/program may be late to catch the next event
18380                                  <1>       ;                within same callback purpose.
18381                                  <1>       ;
18382                                  <1>
18383 000122F2 30C0                   <1>       xor    al, al ; the caller is 'syscalbac' sign/flag
18384 000122F4 E85F180000             <1>       call   set_irq_callback_service
18385                                  <1>       ; 16/04/2017
18386 000122F9 A3[64030300]           <1>       mov    [u.r0], eax
18387 000122FE 0F83C8B4FFFF           <1>       jnc    sysret
18388 00012304 A3[C8030300]           <1>       mov    dword [u.error], eax
18389 00012309 E99EB4FFFF             <1>       jmp    error
18390                                  <1>
18391                                  <1> sysfpstat:
18392                                  <1>       ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
18393                                  <1>       ; (TRDOS 386 feature only!)
18394                                  <1>       ;
18395                                  <1>       ; Set or reset FPU registers save/restore option (for user)
18396                                  <1>       ;             (during software task switching, wswap-rswap)
18397                                  <1>       ;
18398                                  <1>       ; INPUT ->
18399                                  <1>       ;     BL = 0 -> reset
18400                                  <1>       ;     BL = 1 -> set (FPU register will be saved and restored)
18401                                  <1>       ;
18402                                  <1>       ; OUTPUT ->
18403                                  <1>       ;     cf = 0 -> no error, FPU is ready...
18404                                  <1>       ;               (EAX = 0)
18405                                  <1>       ;     Cf = 1 -> error, 80387 FPU is not ready !
18406                                  <1>       ;               (EAX = 0FFFFFFFFh)
18407                                  <1>
18408 0001230E 31C0                   <1>       xor    eax, eax
18409 00012310 803D[AC950100]00       <1>       cmp    byte [fpready], 0
18410 00012317 7613                   <1>       jna    short sysfpstat_err
18411                                  <1>
18412 00012319 80E301                 <1>       and    bl, 1 ; use BIT 0 only !
18413 0001231C 881D[DA030300]         <1>       mov    [u.fpsave], bl
18414 00012322 A3[64030300]           <1>       mov    [u.r0], eax ; 0
18415 00012327 E9A0B4FFFF             <1>       jmp    sysret
18416                                  <1>
18417                                  <1> sysfpstat_err:
18418 0001232C 48                     <1>       dec    eax ; 0FFFFFFFFh
18419 0001232D A3[64030300]           <1>       mov    [u.r0], eax ; -1
18420 00012332 E975B4FFFF             <1>       jmp    error
18421                                  <1>
18422                                  <1> sysdelete: ; Delete (Remove, Unlink) File
18423                                  <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
18424                                  <1>       ;
18425                                  <1>         ; INPUT ->
18426                                  <1>         ;          EBX = File name (ASCIIZ string) address
18427                                  <1>       ; OUTPUT ->
18428                                  <1>       ;          cf = 0 -> eax = 0
18429                                  <1>       ;          cf = 1 -> Error code in AL
18430                                  <1>       ;
18431                                  <1>       ; Modified Registers: EAX (at the return of system call)
18432                                  <1>       ;
18433                                  <1>
18434 00012337 89DE                   <1>       mov    esi, ebx
18435                                  <1>       ; file name is forced, change directory as temporary
18436                                  <1>       ;mov   ax, 1
18437                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18438                                  <1>       ;call  set_working_path
18439 00012339 E8620B0000             <1>       call   set_working_path_x
18440 0001233E 731D                   <1>       jnc    short sysdelete_1
18441                                  <1>
18442 00012340 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
18443 00012342 7505                   <1>       jnz    short sysdelete_err
18444                                  <1>       ; eax = 0
18445                                  <1> sysdelete_path_err:
18446 00012344 B813000000             <1>       mov    eax, ERR_INV_PATH_NAME ; 'bad path name !'
18447                                  <1> sysdelete_err:
18448 00012349 A3[64030300]           <1>       mov    [u.r0], eax
18449 0001234E A3[C8030300]           <1>       mov    [u.error], eax
18450 00012353 E81D0C0000             <1>       call   reset_working_path
18451 00012358 E94FB4FFFF             <1>       jmp    error
```

```
18452                                  <1> sysdelete_1:
18453                                  <1>       ;mov   esi, FindFile_Name
18454 0001235D 66B80018                <1>       mov    ax, 1800h ; Only files
18455 00012361 E8EB6FFFFF              <1>       call   find_first_file
18456 00012366 72E1                    <1>       jc     short sysdelete_err
18457                                  <1> sysdelete_2:
18458                                  <1>       ; check file attributes
18459                                  <1>
18460                                  <1>       ;test  bl, 17 ; system, hidden, readonly, directory
18461 00012368 F6C307                  <1>         test bl, 7 ; system, hidden, readonly
18462 0001236B 7407                    <1>         jz   short sysdelete_3
18463                                  <1>
18464 0001236D B80B000000              <1>         mov  eax, ERR_FILE_ACCESS ; 11 = 'permission denied !'
18465 00012372 EBD5                    <1>         jmp  short sysdelete_err
18466                                  <1> sysdelete_3:
18467 00012374 6621D2                  <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
18468 00012377 7407                    <1>       jz     short sysdelete_4
18469 00012379 B81A000000              <1>       mov    eax, ERR_INV_FILE_NAME ; 26 = 'invalid file name !'
18470 0001237E EBC9                    <1>         jmp  short sysdelete_err
18471                                  <1> sysdelete_4:
18472                                  <1>       ;mov   bh, [LongName_EntryLength]
18473 00012380 883D[1E930100]          <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
18474                                  <1>       ; edi = Directory Entry Offset (DirBuff)
18475                                  <1>       ; esi = Directory Entry (FFF Structure)
18476 00012386 E84399FFFF              <1>       call   remove_file
18477 0001238B 72BC                    <1>       jc     short sysdelete_err
18478                                  <1> sysrmdir_5:
18479 0001238D 31C0                    <1>       xor    eax, eax ; 0
18480 0001238F A3[64030300]            <1>       mov    [u.r0], eax
18481                                  <1>       ;mov   [u.error], eax
18482 00012394 E8DC0B0000              <1>       call   reset_working_path
18483 00012399 E92EB4FFFF              <1>       jmp    sysret
18484                                  <1>
18485                                  <1>
18486                                  <1> sysrmdir: ; Remove (Unlink) Directory
18487                                  <1>       ; 19/01/2021
18488                                  <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
18489                                  <1>       ;
18490                                  <1>         ; INPUT ->
18491                                  <1>         ;          EBX = Pointer to directory name
18492                                  <1>       ; OUTPUT ->
18493                                  <1>       ;          cf = 0 -> eax = 0
18494                                  <1>       ;          cf = 1 -> Error code in AL
18495                                  <1>       ;
18496                                  <1>       ; Modified Registers: EAX (at the return of system call)
18497                                  <1>       ;
18498                                  <1>
18499                                  <1>       ; 19/01/2021
18500 0001239E 803D[B0030300]00        <1>       cmp    byte [u.uid], 0  ; root (super user) ?
18501 000123A5 7614                    <1>       jna    short sysrmdir_0
18502                                  <1>
18503                                  <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
18504 000123A7 B80B000000              <1>       mov    eax, ERR_PERM_DENIED ; ERR_NOT_SUPERUSER
18505 000123AC A3[64030300]            <1>       mov    [u.r0], eax
18506 000123B1 A3[C8030300]            <1>       mov    [u.error], eax
18507 000123B6 E9F1B3FFFF              <1>       jmp    error
18508                                  <1>
18509                                  <1> sysrmdir_0:
18510 000123BB 89DE                    <1>       mov    esi, ebx
18511                                  <1>       ; file name is forced, change directory as temporary
18512                                  <1>       ;mov   ax, 1
18513                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18514                                  <1>       ;call  set_working_path
18515 000123BD E8DE0A0000              <1>       call   set_working_path_x
18516 000123C2 731D                    <1>       jnc    short sysrmdir_1
18517                                  <1>
18518 000123C4 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
18519 000123C6 7505                    <1>       jnz    short sysrmdir_err
18520                                  <1>       ; eax = 0
18521                                  <1> sysrmdir_not_found:
18522 000123C8 B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
18523                                  <1> sysrmdir_err:
18524 000123CD A3[64030300]            <1>       mov    [u.r0], eax
18525 000123D2 A3[C8030300]            <1>       mov    [u.error], eax
18526 000123D7 E8990B0000              <1>       call   reset_working_path
18527 000123DC E9CBB3FFFF              <1>       jmp    error
18528                                  <1> sysrmdir_1:
18529                                  <1>       ;mov   esi, FindFile_Name
18530 000123E1 66B81008                <1>       mov    ax, 0810h ; Only directories
18531 000123E5 E8676FFFFF              <1>       call   find_first_file
18532 000123EA 7306                    <1>       jnc    short sysrmdir_2
18533                                  <1>
18534                                  <1>       ; eax = 2 (File not found !)
18535 000123EC 3C02                    <1>       cmp    al, 2 ; ERR_NOT_FOUND
18536 000123EE 74D8                    <1>       je     short sysrmdir_not_found
18537 000123F0 EBDB                    <1>       jmp    short sysrmdir_err
18538                                  <1> sysrmdir_2:
18539                                  <1>       ; check directory attributes
18540                                  <1>
18541 000123F2 F6C307                  <1>         test bl, 7 ; system, hidden, readonly
18542 000123F5 7407                    <1>         jz   short sysrmdir_3
18543                                  <1>
18544 000123F7 B80B000000              <1>         mov  eax, ERR_DIR_ACCESS ; 11 = 'permission denied !'
18545 000123FC EBCF                    <1>         jmp  short sysrmdir_err
18546                                  <1> sysrmdir_3:
18547 000123FE 6621D2                  <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
18548 00012401 7407                    <1>       jz     short sysrmdir_4
18549                                  <1>       ;mov   eax, ERR_NOT_DIR ; 'not a valid directory !'
18550 00012403 B813000000              <1>       mov    eax, ERR_INV_PATH_NAME ; 'bad path name !'
18551 00012408 EBC3                    <1>         jmp  short sysrmdir_err
18552                                  <1> sysrmdir_4:
18553                                  <1>       ;mov   bh, [LongName_EntryLength]
18554 0001240A 883D[1E930100]          <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
18555                                  <1>       ; edi = Directory Entry Offset (DirBuff)
18556                                  <1>       ; esi = Directory Entry (FFF Structure)
```

```
18557 00012410 E81176FFFF          <1>       call   delete_sub_directory
18558 00012415 0F8372FFFFFF        <1>       jnc    sysrmdir_5
18559                              <1> ;     jc     short sysrmdir_6
18560                              <1> ;
18561                              <1> ;     xor    eax, eax ; 0
18562                              <1> ;sysrmdir_5:
18563                              <1> ;     mov    [u.r0], eax
18564                              <1> ;     ;mov   [u.error], eax
18565                              <1> ;     call   reset_working_path
18566                              <1> ;     jmp    sysret
18567                              <1> sysrmdir_6:
18568 0001241B A3[64030300]        <1>       mov    [u.r0], eax
18569 00012420 A3[C8030300]        <1>       mov    [u.error], eax
18570                              <1>
18571 00012425 09C0                <1>       or     eax, eax ; EAX = 0 -> Directory not empty!
18572 00012427 741C                <1>       jz     short sysrmdir_9
18573                              <1>
18574                              <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
18575                              <1>
18576 00012429 833D[D2900100]01    <1>       cmp    dword [FAT_ClusterCounter], 1
18577 00012430 7209                <1>       jb     short sysrmdir_8
18578                              <1> sysrmdir_7:
18579                              <1>       ; ESI = Logical DOS Drive Description Table address
18580 00012432 66BB00FF            <1>       mov    bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
18581                              <1>                   ; BL = 0 -> Recalculate free cluster count
18582 00012436 E877AEFFFF          <1>       call   calculate_fat_freespace
18583                              <1> sysrmdir_8:
18584 0001243B E8350B0000          <1>       call   reset_working_path
18585 00012440 E967B3FFFF          <1>       jmp    error
18586                              <1>
18587                              <1> sysrmdir_9:
18588 00012445 A1[D2900100]        <1>       mov    eax, [FAT_ClusterCounter]
18589 0001244A 09C0                <1>       or     eax, eax ; 0 ?
18590 0001244C 0F847BFFFFFF        <1>       jz     sysrmdir_err
18591                              <1>       ; ESI = Logical DOS Drive Description Table address
18592 00012452 66BB01FF            <1>       mov    bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
18593                              <1>                   ; BL = 1 -> add free clusters
18594 00012456 E857AEFFFF          <1>       call   calculate_fat_freespace
18595 0001245B 09C9                <1>       or     ecx, ecx
18596 0001245D 74DC                <1>         jz   short sysrmdir_8 ; ecx = 0 -> OK
18597                              <1>       ; ecx > 0 -> Error (Recalculation is needed)
18598 0001245F EBD1                <1>       jmp    short sysrmdir_7
18599                              <1>
18600                              <1>
18601                              <1> syschdir: ; Change Current (Working) Drive & Directory (for user)
18602                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18603                              <1>       ;
18604                              <1>         ; INPUT ->
18605                              <1>         ;          EBX = Directory name (ASCIIZ string) address
18606                              <1>       ; OUTPUT ->
18607                              <1>       ;          cf = 0 -> eax = 0
18608                              <1>       ;          cf = 1 -> Error code in AL
18609                              <1>       ;
18610                              <1>       ; Modified Registers: EAX (at the return of system call)
18611                              <1>       ;
18612                              <1>       ; NOTE: If drive name is not included, only the working
18613                              <1>       ; directory (for user, not for drive/OS) will be chanded.
18614                              <1>       ; If there is a drive name (as A:, B:, C:, D: etc.)
18615                              <1>       ; at the beginning of the ASCIIZ (directory) string,
18616                              <1>       ; working drive and working directory (for user)
18617                              <1>       ; will be changed together.
18618                              <1>       ; (When the program is terminated, MainProg -internal
18619                              <1>       ; shell- will reset working directory to the previous
18620                              <1>       ; -current- logical drive's current directory again.)
18621                              <1>
18622 00012461 89DE                <1>       mov    esi, ebx
18623                              <1>       ; file name is not forced, change directory as temporary
18624 00012463 31C0                <1>       xor    eax, eax
18625                              <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18626                              <1>       ;call  set_working_path
18627 00012465 E83A0A0000          <1>       call   set_working_path_xx
18628 0001246A 731D                <1>       jnc    short syschdir_ok
18629 0001246C 21C0                <1>       and    eax, eax  ; 0 -> Bad Path!
18630 0001246E 7505                <1>       jnz    short syschdir_err
18631                              <1>       ; eax = 0
18632                              <1> syschdir_not_found:
18633 00012470 B80C000000          <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
18634                              <1> syschdir_err:
18635 00012475 A3[64030300]        <1>       mov    [u.r0], eax
18636 0001247A A3[C8030300]        <1>       mov    [u.error], eax
18637 0001247F E8F10A0000          <1>       call   reset_working_path
18638 00012484 E923B3FFFF          <1>       jmp    error
18639                              <1> syschdir_ok:
18640 00012489 31C0                <1>       xor    eax, eax ; 0
18641 0001248B A3[64030300]        <1>       mov    [u.r0], eax
18642                              <1>       ;mov   [u.error], eax
18643 00012490 E937B3FFFF          <1>       jmp    sysret
18644                              <1>
18645                              <1>
18646                              <1> syschmod: ; Get & Change File (or Directory) Attributes
18647                              <1>       ; 19/01/2021
18648                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18649                              <1>       ;
18650                              <1>         ; INPUT ->
18651                              <1>         ;          EBX = File/Directory (ASCIIZ) name address
18652                              <1>       ;          CL = New attributes (if CL < 40h)
18653                              <1>       ;          CL >= 40h -> Get File Attributes
18654                              <1>       ; OUTPUT ->
18655                              <1>       ;          cf = 0 -> EAX = File attributes (in AL)
18656                              <1>       ;          cf = 1 -> Error code in AL
18657                              <1>       ;
18658                              <1>       ; Modified Registers: EAX (at the return of system call)
18659                              <1>       ;
18660                              <1>       ; MSDOS File Attributes:    (bit value of attrib byte)
18661                              <1>       ;     ATTR_READ_ONLY      =     01h  (bit 0, 'R')
```

```
18662                                  <1>       ;      ATTR_HIDDEN  =      02h  (bit 1, 'H')
18663                                  <1>       ;      ATTR_SYSTEM  =      04h  (bit 2, 'S')
18664                                  <1>       ;      ATTR_VOLUME_ID      =      08h  (bit 3)
18665                                  <1>       ;      ATTR_DIRECTORY      =      10h  (bit 4)
18666                                  <1>       ;      ATTR_ARCHIVE =      20h  (bit 5, 'A')
18667                                  <1>       ;      ATTR_LONG_NAME      =      ATTR_READONLY |
18668                                  <1>       ;                          ATTR_HIDDEN |
18669                                  <1>       ;                          ATTR_SYSTEM |
18670                                  <1>       ;                          ATTR_VOLUME_ID
18671                                  <1>       ;      The upper two bits of attributes must be 0.
18672                                  <1>
18673                                  <1>       ; Note:      * If ATTR_DIRECTORY is set, only directory names
18674                                  <1>       ;              will be searched (and S,H,R,A attributeds of
18675                                  <1>       ;              the directory will be changed.)
18676                                  <1>       ;            * If ATTR_VOLUME_ID is set, 'syschmod' system call
18677                                  <1>       ;              will return with 'permission denied' error.
18678                                  <1>       ;            * If ATTR_DIRECTORY is not set, only file names
18679                                  <1>       ;              will be searched (and S,H,R,A attributes of the
18680                                  <1>       ;              file will be changed.)
18681                                  <1>       ;
18682                                  <1>       ; (Ony Super User can change S,H,R attributes.)
18683                                  <1>
18684 00012495 80F940                 <1>       cmp    cl, 40h
18685 00012498 7327                   <1>       jnb    short syschmod_0
18686                                  <1>
18687 0001249A F6C108                 <1>       test   cl, 08h ; ATTR_VOLUME_ID
18688 0001249D 750E                   <1>       jnz    short syschmod_perm_err
18689                                  <1>
18690                                  <1>       ; 19/01/2021
18691 0001249F 803D[B0030300]00       <1>       cmp    byte [u.uid], 0  ; root (super user) ?
18692 000124A6 7619                   <1>       jna    short syschmod_0
18693                                  <1>
18694                                  <1>       ; Not super user..
18695 000124A8 F6C107                 <1>       test   cl, 07h      ; S,H,R attributes
18696 000124AB 7414                   <1>       jz     short syschmod_0
18697                                  <1>
18698                                  <1> syschmod_perm_err:
18699                                  <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
18700 000124AD B80B000000             <1>       mov    eax, ERR_PERM_DENIED ; 'permission denied !'
18701 000124B2 A3[64030300]           <1>       mov    [u.r0], eax
18702 000124B7 A3[C8030300]           <1>       mov    [u.error], eax
18703 000124BC E9EBB2FFFF             <1>       jmp    error
18704                                  <1>
18705                                  <1> syschmod_0:
18706 000124C1 880D[6C930100]         <1>       mov    [Attributes], cl
18707 000124C7 89DE                   <1>       mov    esi, ebx
18708                                  <1>       ; file name is forced, change directory as temporary
18709                                  <1>       ;mov   ax, 1
18710                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18711                                  <1>       ;call  set_working_path
18712 000124C9 E8D2090000             <1>       call   set_working_path_x
18713 000124CE 731D                   <1>       jnc    short syschmod_1
18714 000124D0 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
18715 000124D2 7505                   <1>       jnz    short syschmod_err
18716                                  <1>       ; eax = 0
18717                                  <1> syschmod_path_not_found:
18718 000124D4 B813000000             <1>       mov    eax, ERR_INV_PATH_NAME ; 'Bad path name !'
18719                                  <1> syschmod_err:
18720 000124D9 A3[64030300]           <1>       mov    [u.r0], eax
18721 000124DE A3[C8030300]           <1>       mov    [u.error], eax
18722 000124E3 E88D0A0000             <1>       call   reset_working_path
18723 000124E8 E9BFB2FFFF             <1>       jmp    error
18724                                  <1> syschmod_1:
18725 000124ED B008                   <1>       mov    al, 08h ; Except volume labels (& long names)
18726 000124EF A0[6C930100]           <1>       mov    al, [Attributes]
18727 000124F4 2410                   <1>       and    al, 10h ;
18728                                  <1>       ;mov   esi, FindFile_Name
18729                                  <1>       ;mov   ax, 1800h ; Only files
18730                                  <1>       ;mov   ax, 0810h ; Only directories
18731 000124F6 E8566EFFFF             <1>       call   find_first_file
18732                                  <1>       ;jnc   short syschmod_2
18733 000124FB 72DC                   <1>       jc     short syschmod_err
18734                                  <1>
18735                                  <1>       ;; eax = 2 (File not found !)
18736                                  <1>       ;cmp   al, 2 ; ERR_NOT_FOUND
18737                                  <1>       ;jne   short syschmod_err
18738                                  <1>
18739                                  <1>       ;and   byte [Attributes], 10h
18740                                  <1>       ;jz    short syschmod_err
18741                                  <1>
18742                                  <1>       ;; Directory not found !
18743                                  <1>       ;mov   al, 3 ; ERR_PATH_NOT_FOUND
18744                                  <1>       ;jmp   short syschmod_err
18745                                  <1>
18746                                  <1> syschmod_2:
18747 000124FD 6621D2                 <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
18748 00012500 7407                   <1>       jz     short syschmod_3
18749 00012502 B81A000000             <1>       mov    eax, ERR_INV_FILE_NAME ; 'invalid file name !'
18750 00012507 EBD0                   <1>         jmp  short syschmod_err
18751                                  <1> syschmod_3:
18752                                  <1>       ; EDI = Directory buffer entry offset/address
18753                                  <1>       ; BL = File (or Directory) Attributes
18754                                  <1>       ; mov  bl, [EDI+0Bh]
18755                                  <1>
18756                                  <1>       ; check directory attributes
18757 00012509 8A3D[6C930100]         <1>       mov    bh, [Attributes] ; new attributes
18758 0001250F 80FF40                 <1>       cmp    bh, 40h  ;>=40 -> get file/directory attributes
18759 00012512 732D                   <1>       jnb    short syschmod_6
18760                                  <1>
18761                                  <1>       ; set file/directory attributes
18762 00012514 F6C307                 <1>       test   bl, 7 ; system, hidden, readonly
18763 00012517 7409                   <1>         jz   short syschmod_4
18764                                  <1>
18765                                  <1>       ; 19/01/2021
18766 00012519 803D[B0030300]00       <1>       cmp    byte [u.uid], 0  ; root (super user) ?
```

```
18767 00012520 778B              <1>       ja     short syschmod_perm_err
18768                            <1> syschmod_4:
18769 00012522 66817F0CA101      <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
18770 00012528 7424              <1>       je     short syschmod_7
18771                            <1>
18772 0001252A 887F0B            <1>       mov    [edi+0Bh], bh    ; Attributes (New!)
18773                            <1>
18774 0001252D C605[DC900100]02  <1>       mov    byte [DirBuff_ValidData], 2 ; modified sign
18775                            <1>                                   ; to force write
18776 00012534 E84D94FFFF        <1>       call   save_directory_buffer
18777 00012539 729E              <1>       jc     short syschmod_err
18778                            <1>
18779                            <1> syschmod_5:
18780 0001253B 8A1D[6C930100]    <1>       mov    bl, [Attributes]
18781                            <1> syschmod_6:
18782 00012541 0FB6C3            <1>       movzx  eax, bl
18783 00012544 A3[64030300]      <1>       mov    [u.r0], eax
18784                            <1>       ;mov   dword [u.error], 0
18785 00012549 E97EB2FFFF        <1>       jmp    sysret
18786                            <1>
18787                            <1> syschmod_7:
18788 0001254E 29C0              <1>       sub    eax, eax
18789 00012550 8A25[DA900100]    <1>         mov     ah, [DirBuff_DRV]
18790 00012556 BE00010900        <1>       mov    esi, Logical_DOSDisks
18791 0001255B 01C6              <1>         add     esi, eax
18792 0001255D 807E04A1          <1>         cmp     byte [esi+LD_FSType], 0A1h
18793 00012561 7307              <1>       jnc    short syschmod_8
18794 00012563 B01D              <1>       mov    al, ERR_INV_DATA ; 29 = Invalid Data
18795 00012565 E96FFFFFFF        <1>       jmp    syschmod_err
18796                            <1>
18797                            <1> syschmod_8:
18798                            <1>       ; BH = New MS-DOS File Attributes
18799 0001256A 88F8              <1>       mov    al, bh ; File/Directory Attributes
18800 0001256C 30E4              <1>       xor    ah, ah ; Attributes in MS-DOS format sign
18801 0001256E E83C7FFFFF        <1>       call   change_fs_file_attributes
18802 00012573 0F8260FFFFFF      <1>       jc     syschmod_err
18803 00012579 EBC0              <1>       jmp    short syschmod_5
18804                            <1>
18805                            <1>
18806                            <1> sysdrive: ; Get/Set Current (Working) Drive (for user)
18807                            <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18808                            <1>       ;
18809                            <1>         ; INPUT ->
18810                            <1>         ;          BL = Logical DOS Drive number (0=A: ... 2=C:)
18811                            <1>       ;         If BL = 0FFh -> Get Current Drive
18812                            <1>       ; OUTPUT ->
18813                            <1>       ;          cf = 0 ->
18814                            <1>       ;               AL = Current Drive number
18815                            <1>       ;               AH = The Last Logical DOS Drive no.
18816                            <1>       ;          cf = 1 -> Error code in AL
18817                            <1>       ;
18818                            <1>       ; Modified Registers: EAX (at the return of system call)
18819                            <1>       ;
18820                            <1>       ; NOTE: If the requested logical dos drive is ready,
18821                            <1>       ;     it's current current directory will be the user's
18822                            <1>       ;     (program's) current directory.
18823                            <1>       ;     (When the program is terminated, MainProg -internal
18824                            <1>       ;     shell- will reset the previous -current- logical drive
18825                            <1>       ;       as current drive again).
18826                            <1>
18827 0001257B 80FBFF            <1>       cmp    bl, 0FFh
18828 0001257E 7435              <1>       je     short sysdrive_ok
18829 00012580 3A1D[823F0100]    <1>       cmp    bl, [Last_DOS_DiskNo]
18830 00012586 771E              <1>       ja     short sysdrive_err
18831                            <1>
18832                            <1>       ; Save current drive and reset mode
18833                            <1>       ; for 'reset_working_path' procedure (for MainProg)
18834 00012588 30C0              <1>       xor    al, al
18835 0001258A 66A3[A8950100]    <1>       mov    [SWP_Mode], ax ; ah = 0
18836 00012590 A0[B6890100]      <1>       mov    al, [Current_Drv]
18837 00012595 FEC4              <1>       inc    ah ; mov ah, 1
18838 00012597 66A3[AA950100]    <1>       mov    [SWP_DRV], ax
18839                            <1>
18840 0001259D 88DA              <1>       mov    dl, bl
18841 0001259F E80A5AFFFF        <1>       call   change_current_drive
18842 000125A4 730F              <1>       jnc    short sysdrive_ok
18843                            <1> sysdrive_err:
18844 000125A6 C705[64030300]0F00- <1>     mov    dword [u.r0], ERR_DRV_NOT_RDY ; 'drive not ready !'
18844 000125AE 0000              <1>
18845 000125B0 E9F7B1FFFF        <1>       jmp    error
18846                            <1> sysdrive_ok:
18847 000125B5 A0[B6890100]      <1>       mov    al, [Current_Drv]
18848 000125BA 8A25[823F0100]    <1>       mov    ah, [Last_DOS_DiskNo]
18849 000125C0 A3[64030300]      <1>       mov    [u.r0], eax
18850 000125C5 E902B2FFFF        <1>       jmp    sysret
18851                            <1>
18852                            <1>
18853                            <1> sysdir: ; Get Current (Working) Drive & Directory (for user)
18854                            <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18855                            <1>       ;
18856                            <1>         ; INPUT ->
18857                            <1>         ;          EBX = Current directory name buffer address
18858                            <1>       ;            (Buffer length = 92 bytes)
18859                            <1>       ; OUTPUT ->
18860                            <1>       ;          AL = Current drive (0=A: .. 2=C:)
18861                            <1>       ;         If CF = 1 -> AL = error code
18862                            <1>       ;
18863                            <1>       ; Modified Registers: EAX (at the return of system call)
18864                            <1>       ;
18865                            <1>       ; Note: Required directory name buffer length may be
18866                            <1>       ;     <= 92 bytes for current TRDOS 386 version.
18867                            <1>       ;     (7*12 name chars + 7 slash + 0)
18868                            <1>
18869 000125CA 89E5              <1>       mov    ebp, esp
18870 000125CC 83EC60            <1>       sub    esp, 96
```

```
18871 000125CF 53                  <1>       push   ebx ; User's buffer address
18872 000125D0 30D2                <1>       xor    dl, dl ; 0 = current drive
18873 000125D2 E8D388FFFF          <1>       call   get_current_directory
18874 000125D7 72CD                <1>       jc     short sysdrive_err ; 'drive not ready !' error
18875 000125D9 89E6                <1>       mov    esi, esp ; System's buffer address
18876 000125DB 5F                  <1>       pop    edi  ; User's buffer address
18877                              <1>       ; ecx = transfer (byte) count (<=92)
18878 000125DC E843F4FFFF          <1>       call   transfer_to_user_buffer
18879 000125E1 89EC                <1>       mov    esp, ebp
18880 000125E3 730F                <1>       jnc    short sysdir_ok
18881                              <1> sysdir_err:
18882 000125E5 C705[64030300]2E00- <1>       mov    dword [u.r0], ERR_BUFFER  ; 'buffer error !'
18882 000125ED 0000                <1>
18883 000125EF E9B8B1FFFF          <1>       jmp    error
18884                              <1> sysdir_ok:
18885 000125F4 8A0D[B6890100]      <1>       mov    cl, [Current_Drv]
18886 000125FA 890D[64030300]      <1>       mov    [u.r0], ecx
18887 00012600 E9C7B1FFFF          <1>       jmp    sysret
18888                              <1>
18889                              <1>
18890                              <1> sysldrvt: ; Get copy of Logical DOS Drive Description Table
18891                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18892                              <1>       ;
18893                              <1>         ; INPUT ->
18894                              <1>       ;          BL = Logical DOS drive number (zero based)
18895                              <1>         ;           ECX = Logical DOS drv desc table buffer addr
18896                              <1>       ;             (Buffer length = 256 bytes)
18897                              <1>       ; OUTPUT ->
18898                              <1>       ;          cf = 0 ->
18899                              <1>       ;               AL = Current Drive number
18900                              <1>       ;               AH = The Last Logical DOS Drive no.
18901                              <1>       ;          cf = 1 -> Error code in AL
18902                              <1>       ;               AH = The Last Logical DOS Drive no.
18903                              <1>       ;
18904                              <1>       ; Modified Registers: EAX (at the return of system call)
18905                              <1>       ;
18906                              <1>       ; Note: Required description table buffer length is
18907                              <1>       ;     256 bytes for current TRDOS 386 version.
18908                              <1>
18909 00012605 89CF                <1>       mov    edi, ecx ; Destination address (user space)
18910 00012607 88DC                <1>       mov    ah, bl
18911 00012609 30C0                <1>       xor    al, al
18912 0001260B BE00010900          <1>       mov    esi, Logical_DOSDisks
18913 00012610 01C6                <1>       add    esi, eax ; Source address (system space)
18914 00012612 B900010000          <1>       mov    ecx, 256 ; Byte count
18915                              <1>                       ; Logical Dos Drv Desc Table size
18916 00012617 E808F4FFFF          <1>       call   transfer_to_user_buffer
18917 0001261C 72C7                <1>       jc     short sysdir_err
18918 0001261E 8A2D[823F0100]      <1>       mov    ch, [Last_DOS_DiskNo]
18919 00012624 EBCE                <1>       jmp    short sysdir_ok
18920                              <1>
18921                              <1>
18922                              <1> systime: ; Get System Date&Time
18923                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18924                              <1>       ;
18925                              <1>         ; INPUT -> BL =
18926                              <1>       ;          0 = Get Date&Time in Unix/Epoch format
18927                              <1>       ;          1 = Get Time in MSDOS format
18928                              <1>       ;          2 = Get Date in MSDOS format
18929                              <1>       ;          3 = Get Date&Time in MSDOS format
18930                              <1>       ;          4 & other values =
18931                              <1>       ;             System timer ticks will be returned
18932                              <1>       ;             in EAX and Carry Flag will be set.
18933                              <1>       ;             (CF will not be set if BL = 4)
18934                              <1>       ; OUTPUT ->
18935                              <1>       ;      For BL input = 3
18936                              <1>       ;          EAX = Current Time (RTC)
18937                              <1>       ;            AL = Second (DL in MSDOS)
18938                              <1>       ;            AH = Minute (CL in MSDOS)
18939                              <1>       ;            HW of EAX = Hour (CH in MSDOS)
18940                              <1>       ;         EDX = Current System Date (RTC)
18941                              <1>       ;            DL = Day (DL in MSDOS)
18942                              <1>       ;            DH = Month (DH in MSDOS)
18943                              <1>       ;            HW of EDX = Year (CX in MSDOS)
18944                              <1>       ;
18945                              <1>       ;      For BL input = 2
18946                              <1>       ;         EAX = Current System Date (RTC)
18947                              <1>       ;            DL = Day (DL in MSDOS)
18948                              <1>       ;            DH = Month (DH in MSDOS)
18949                              <1>       ;            HW of EDX = Year (CX in MSDOS)
18950                              <1>       ;
18951                              <1>       ;      For BL input = 1
18952                              <1>       ;          EAX = Current Time (RTC)
18953                              <1>       ;            AL = Second (DL in MSDOS)
18954                              <1>       ;            AH = Minute (CL in MSDOS)
18955                              <1>       ;            HW of EAX = Hour (CH in MSDOS)
18956                              <1>       ;
18957                              <1>       ;      For BL input = 0
18958                              <1>       ;          EAX = Unix (Epoch) Time Ticks/Seconds
18959                              <1>       ;
18960                              <1>       ;      For BL input  = 4
18961                              <1>       ;         EAX = System timer ticks
18962                              <1>       ;
18963                              <1>       ;      If CF = 1 (for other values of BL input)
18964                              <1>       ;         EAX = System timer ticks (no error code!)
18965                              <1>       ;
18966                              <1>       ; Modified Registers: EAX, (EDX)
18967                              <1>       ;             (at the return of system call)
18968                              <1>       ;
18969                              <1>
18970 00012626 20DB                <1>       and    bl, bl
18971 00012628 750F                <1>       jnz    short systime_1
18972 0001262A E86F4FFFFF          <1>       call   epoch
18973                              <1> systime_0:
18974 0001262F A3[64030300]        <1>       mov    [u.r0], eax
```

```
18975 00012634 E993B1FFFF          <1>       jmp    sysret
18976                              <1> systime_1:
18977 00012639 80FB04              <1>       cmp    bl, 4
18978 0001263C 7211                <1>       jb     short systime_2
18979 0001263E A1[70890100]        <1>       mov    eax, [TIMER_LH] ; 18.2 Hz timer ticks
18980                              <1>                               ; Note: [TIMER_LH] may be set
18981                              <1>                               ; to wrong timer value due to
18982                              <1>                               ; program functions.
18983                              <1>                               ; (This value must not be
18984                              <1>                               ; accepted as [TIMER_LH]/18.2
18985                              <1>                               ; seconds since the midnight.)
18986 00012643 76EA                <1>       jna    short systime_0
18987 00012645 A3[64030300]        <1>       mov    [u.r0], eax
18988 0001264A E95DB1FFFF          <1>       jmp    error ; cf = 1 & [u.r0] = eax = timer ticks
18989                              <1>
18990                              <1> systime_2:
18991                              <1>       ;push  ebx
18992 0001264F E8AC4EFFFF          <1>       call   get_rtc_date_time
18993                              <1>       ;pop   ebx
18994 00012654 F6C301              <1>       test   bl, 1
18995 00012657 7429                <1>       jz     short systime_4
18996 00012659 30E4                <1>       xor    ah, ah
18997 0001265B A0[BE850100]        <1>       mov    al, [hour]
18998 00012660 88C2                <1>       mov    dl, al
18999 00012662 C1E010              <1>       shl    eax, 16
19000 00012665 A0[C2850100]        <1>       mov    al, [second]
19001 0001266A 8A25[C0850100]      <1>       mov    ah, [minute]
19002 00012670 F6C302              <1>       test   bl, 2
19003 00012673 74BA                <1>       jz     short systime_0
19004                              <1>       ; Check time & date match risk
19005                              <1>       ; (23:59:59 may cause to wrong
19006                              <1>       ; date -new day with previous date-...)
19007 00012675 80FA17              <1>       cmp    dl, 23
19008 00012678 7206                <1>       jb     short systime_3
19009 0001267A 663D3B3B            <1>       cmp    ax, (59*256)+59 ; if hour is 23:59:59
19010 0001267E 73CF                <1>       jnb    short systime_2 ; wait for 1 second
19011                              <1> systime_3:
19012                              <1>       ; eax = time
19013 00012680 89C6                <1>       mov    esi, eax
19014                              <1> systime_4:
19015 00012682 66A1[B8850100]      <1>       mov    ax, [year]
19016 00012688 C1E010              <1>       shl    eax, 16
19017 0001268B A0[BC850100]        <1>       mov    al, [day]
19018 00012690 8A25[BA850100]      <1>       mov    ah, [month]
19019                              <1>       ; eax = date
19020 00012696 80E301              <1>       and    bl, 1
19021 00012699 7494                <1>       jz     short systime_0
19022 0001269B 96                  <1>       xchg   esi, eax
19023                              <1>       ; eax = time, esi = date
19024 0001269C 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; EBP points to user's registers
19025                              <1>       ; (user) edx <-- (system) esi
19026 000126A2 897514              <1>       mov    [ebp+20], esi ; return to user with EDX value
19027 000126A5 EB88                <1>       jmp    short systime_0
19028                              <1>
19029                              <1>
19030                              <1> sysstime: ; Set System Date&Time
19031                              <1>       ; 31/12/2017
19032                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
19033                              <1>       ;
19034                              <1>         ; INPUT -> BL =
19035                              <1>       ;          0 = Set Date&Time in Unix/Epoch format
19036                              <1>       ;          1 = Set Time in MSDOS format
19037                              <1>       ;          2 = Set Date in MSDOS format
19038                              <1>       ;          3 = Set Date&Time in MSDOS format
19039                              <1>       ;          4 = Set System Timer (Ticks)
19040                              <1>       ;          5 = Convert/Save current time to/as
19041                              <1>       ;             18.2 Hz system timer ticks
19042                              <1>       ;          6 = Convert MSDOS Date&Time to UNIX format
19043                              <1>       ;             without setting system date&time ; (test)
19044                              <1>       ;          7 = Convert UNIX Date&Time to MSDOS format
19045                              <1>       ;             without setting system date&time ; (test)
19046                              <1>       ;         8-0FFh = invalid !
19047                              <1>       ;         ECX = Time (or Timer) value in selected format
19048                              <1>       ;         EDX = Date value in MSDOS format if BL=2,3,6
19049                              <1>       ;
19050                              <1>       ; OUTPUT ->
19051                              <1>       ;      If CF = 0 ->
19052                              <1>       ;           EAX = Set value
19053                              <1>       ;      If CF = 1 -> (invalid BL input)
19054                              <1>       ;         EAX = Ticks count [TIMER_LH]
19055                              <1>       ;
19056                              <1>
19057 000126A7 20DB                <1>       and    bl, bl ; 0
19058 000126A9 7511                <1>       jnz    short sysstime_0
19059 000126AB 89C8                <1>       mov    eax, ecx
19060 000126AD E8764FFFFF          <1>       call   convert_from_epoch
19061 000126B2 E82250FFFF          <1>       call   set_rtc_date_time
19062 000126B7 E910B1FFFF          <1>       jmp    sysret
19063                              <1> sysstime_0:
19064 000126BC 80FB08              <1>       cmp    bl, 8
19065 000126BF 722D                <1>       jb     short sysstime_1
19066                              <1>       ; invalid input (>7)
19067 000126C1 A1[70890100]        <1>       mov    eax, [TIMER_LH] ; 18.2 Hz timer ticks
19068                              <1>                               ; Note: [TIMER_LH] may be set
19069                              <1>                               ; to wrong timer value due to
19070                              <1>                               ; program functions.
19071                              <1>                               ; (This value must not be
19072                              <1>                               ; accepted as [TIMER_LH]/18.2
19073                              <1>                               ; seconds since the midnight.)
19074 000126C6 A3[64030300]        <1>       mov    [u.r0], eax
19075 000126CB E9DCB0FFFF          <1>       jmp    error ; cf = 1 & [u.r0] = eax = timer ticks
19076                              <1>
19077                              <1> sysstime_8:
19078                              <1>       ; BL = 7
19079 000126D0 89C8                <1>       mov    eax, ecx ; seconds since 1/1/1970 00:00:00
```

```
19080 000126D2 E8514FFFFF          <1>       call   convert_from_epoch
19081 000126D7 30E4                <1>       xor    ah, ah
19082 000126D9 A0[BE850100]        <1>       mov    al, [hour]
19083 000126DE C1E010              <1>       shl    eax, 16
19084 000126E1 A0[C2850100]        <1>       mov    al, [second]
19085 000126E6 8A25[C0850100]      <1>       mov    ah, [minute]
19086 000126EC EB92                <1>       jmp    short systime_3
19087                              <1>
19088                              <1> sysstime_1:
19089 000126EE 80FB04              <1>       cmp    bl, 4
19090 000126F1 743F                <1>       je     short sysstime_2 ; set system timer ticks
19091 000126F3 80FB05              <1>       cmp    bl, 5
19092 000126F6 754B                <1>       jne    short sysstime_4
19093                              <1>       ; convert current time to system timer ticks (18.2Hz)
19094 000126F8 E8034EFFFF          <1>       call   get_rtc_date_time
19095 000126FD 0FB60D[BE850100]    <1>       movzx  ecx, byte [hour]
19096 00012704 B8100E0000          <1>       mov    eax, 60*60 ; 1 hour = 3600 seconds
19097 00012709 F7E1                <1>       mul    ecx
19098 0001270B 89C3                <1>       mov    ebx, eax
19099 0001270D B13C                <1>       mov    cl, 60  ; 1 minute = 60 seconds
19100 0001270F 0FB605[C0850100]    <1>       movzx  eax, byte [minute]
19101 00012716 F7E1                <1>       mul    ecx
19102 00012718 01D8                <1>       add    eax, ebx
19103 0001271A 8A0D[C2850100]      <1>       mov    cl, [second]
19104 00012720 01C8                <1>       add    eax, ecx
19105 00012722 B1B6                <1>       mov    cl, 182
19106 00012724 F7E1                <1>       mul    ecx
19107 00012726 83C009              <1>       add    eax, 9
19108 00012729 83D200              <1>       adc    edx, 0
19109 0001272C B10A                <1>       mov    cl, 10
19110 0001272E F7F1                <1>       div    ecx
19111                              <1>       ; eax = ((182*seconds)+9)/10
19112 00012730 89C1                <1>       mov    ecx, eax
19113                              <1> sysstime_2:
19114 00012732 890D[70890100]      <1>       mov    [TIMER_LH], ecx ; 18.2 * seconds
19115                              <1> sysstime_3:
19116 00012738 890D[64030300]      <1>       mov    [u.r0], ecx
19117 0001273E E989B0FFFF          <1>       jmp    sysret
19118                              <1> sysstime_4:
19119 00012743 80FB06              <1>       cmp    bl, 6
19120 00012746 7788                <1>       ja     short sysstime_8
19121                              <1>
19122 00012748 890D[64030300]      <1>       mov    [u.r0], ecx
19123                              <1>
19124 0001274E 880D[C2850100]      <1>       mov    [second], cl
19125 00012754 882D[C0850100]      <1>       mov    [minute], ch
19126 0001275A C1E910              <1>       shr    ecx, 16
19127 0001275D 880D[BE850100]      <1>       mov    [hour], cl
19128                              <1>       ; BL = 1,2,3,6
19129 00012763 80FB01              <1>       cmp    bl, 1
19130 00012766 762A                <1>       jna    short sysstime_5
19131                              <1>       ; BL = 2,3,6
19132 00012768 8815[BC850100]      <1>       mov    [day], dl
19133 0001276E 8835[BA850100]      <1>       mov    [month], dh
19134 00012774 C1EA10              <1>       shr    edx, 16
19135 00012777 668915[B8850100]    <1>       mov    [year], dx
19136 0001277E 80E303              <1>       and    bl, 3
19137 00012781 742D                <1>       jz     short sysstime_7 ; 6
19138                              <1>       ; BL = 2,3
19139 00012783 F6C301              <1>       test   bl, 1
19140 00012786 7419                <1>       jz     short sysstime_6 ; 2
19141                              <1>       ; BL = 3
19142 00012788 E84C4FFFFF          <1>       call   set_rtc_date_time
19143 0001278D E93AB0FFFF          <1>       jmp    sysret
19144                              <1> sysstime_5:
19145                              <1>       ; BL = 1
19146 00012792 E8834FFFFF          <1>       call   set_time_bcd
19147 00012797 E81142FFFF          <1>       call   set_rtc_time
19148 0001279C E92BB0FFFF          <1>       jmp    sysret
19149                              <1> sysstime_6:
19150                              <1>       ; BL = 2
19151 000127A1 E8474FFFFF          <1>       call   set_date_bcd
19152 000127A6 E87142FFFF          <1>       call   set_rtc_date
19153 000127AB E91CB0FFFF          <1>       jmp    sysret
19154                              <1> sysstime_7:
19155                              <1>       ; BL = 6
19156                              <1>       ; [year], [month], [day],
19157                              <1>       ; [hour], [minute], [second]
19158 000127B0 E8EE4DFFFF          <1>       call   convert_to_epoch
19159 000127B5 89C1                <1>       mov    ecx, eax ; seconds since 1/1/1970 00:00:00
19160 000127B7 E97CFFFFFF          <1>       jmp    sysstime_3
19161                              <1>
19162                              <1> sysrename: ; Rename File (or Directory)
19163                              <1>       ; 19/01/2021
19164                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19165                              <1>       ;
19166                              <1>         ; INPUT ->
19167                              <1>         ;          EBX = File/Directory (ASCIIZ) name address
19168                              <1>       ;         ECX = New name (in same dir, no path name)
19169                              <1>       ; OUTPUT ->
19170                              <1>       ;          cf = 0 -> EAX = 0
19171                              <1>       ;          cf = 1 -> Error code in AL
19172                              <1>
19173                              <1>       ; 19/01/2021
19174 000127BC 803D[B0030300]00    <1>       cmp    byte [u.uid], 0  ; root (super user) ?
19175 000127C3 7614                <1>       jna    short sysrename_0
19176                              <1>
19177                              <1> sysrename_perm_err:
19178                              <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
19179 000127C5 B80B000000          <1>       mov    eax, ERR_PERM_DENIED ; 'permission denied !'
19180 000127CA A3[64030300]        <1>       mov    [u.r0], eax
19181 000127CF A3[C8030300]        <1>       mov    [u.error], eax
19182 000127D4 E9D3AFFFFF          <1>       jmp    error
19183                              <1>
19184                              <1> sysrename_0:
```

```
19185 000127D9 51                <1>       push   ecx ; new file name address (in user space)
19186 000127DA 89DE              <1>       mov    esi, ebx
19187                            <1>       ; file name is forced, change directory as temporary
19188                            <1>       ;mov   ax, 1
19189                            <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
19190                            <1>       ;call  set_working_path
19191 000127DC E8BF060000        <1>       call   set_working_path_x
19192 000127E1 731E              <1>       jnc    short sysrename_1
19193 000127E3 21C0              <1>       and    eax, eax  ; 0 -> Bad Path!
19194 000127E5 7505              <1>       jnz    short sysrename_err
19195                            <1>       ; eax = 0
19196                            <1> sysrename_path_not_found:
19197 000127E7 B813000000        <1>       mov    eax, ERR_INV_PATH_NAME ; 'Bad path name !'
19198                            <1> sysrename_err:
19199 000127EC 59                <1>       pop    ecx ; new file name address (in user space)
19200                            <1> sysrename_error:
19201 000127ED A3[64030300]      <1>       mov    [u.r0], eax
19202 000127F2 A3[C8030300]      <1>       mov    [u.error], eax
19203 000127F7 E879070000        <1>       call   reset_working_path
19204 000127FC E9ABAFFFFF        <1>       jmp    error
19205                            <1> sysrename_1:
19206 00012801 B008              <1>       mov    al, 08h ; Except volume labels (& long names)
19207 00012803 A0[6C930100]      <1>       mov    al, [Attributes]
19208 00012808 2410              <1>       and    al, 10h ;
19209                            <1>       ;mov   esi, FindFile_Name
19210                            <1>       ;mov   ax, 1800h ; Only files
19211                            <1>       ;mov   ax, 0810h ; Only directories
19212 0001280A 66B80008          <1>       mov    ax, 0800h ; Find File or Directory
19213 0001280E E83E6BFFFF        <1>       call   find_first_file
19214                            <1>       ;jnc   short sysrename_2
19215 00012813 72D7              <1>       jc     short sysrename_err
19216                            <1> sysrename_2:
19217                            <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
19218                            <1>       ; EDI = Directory Buffer Directory Entry Location
19219                            <1>       ; EAX = File Size
19220                            <1>       ;  BL = Attributes of The File/Directory
19221                            <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
19222                            <1>       ;  DX > 0 : Ambiguous filename chars are used
19223                            <1>
19224 00012815 6621D2            <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
19225 00012818 7407              <1>       jz     short sysrename_3
19226 0001281A B81A000000        <1>       mov    eax, ERR_INV_FILE_NAME ; 'invalid file name !'
19227 0001281F EBCB              <1>         jmp  short sysrename_err
19228                            <1> sysrename_3:
19229                            <1>       ; EDI = Directory buffer entry offset/address
19230                            <1>       ; BL = File (or Directory) Attributes
19231                            <1>       ; mov  bl, [EDI+0Bh]
19232                            <1>
19233 00012821 5A                <1>       pop    edx ; new file name address (in user space)
19234                            <1>
19235                            <1>       ; check file/directory attributes
19236 00012822 F6C307            <1>       test   bl, 7 ; system, hidden, readonly
19237 00012825 759E              <1>         jnz  short sysrename_perm_err
19238                            <1> sysrename_4:
19239 00012827 66817F0CA101      <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
19240 0001282D 7496              <1>       je     short sysrename_perm_err ; -temporary!-
19241                            <1>
19242                            <1>       ; save old file name & file info (FFF structure)
19243 0001282F BE[56920100]      <1>       mov    esi, FindFile_Drv
19244 00012834 BF[9C930100]      <1>       mov    edi, SourceFile_Drv
19245 00012839 B920000000        <1>       mov    ecx, 128/4
19246 0001283E F3A5              <1>       rep    movsd
19247                            <1>
19248 00012840 89D6              <1>       mov    esi, edx ; new file name address (in user space)
19249 00012842 BF[1C940100]      <1>       mov    edi, DestinationFile_Drv
19250 00012847 E8D68CFFFF        <1>       call   parse_path_name
19251 0001284C 729F              <1>       jc     short sysrename_error ; eax = 1 (Bad file name)
19252                            <1>
19253                            <1>       ; same drive ?
19254 0001284E A0[56920100]      <1>       mov    al, [FindFile_Drv]
19255 00012853 3A05[1C940100]    <1>       cmp    al, [DestinationFile_Drv]
19256                            <1>       ;jne   short sysrename_perm_err ; Permission denied
19257 00012859 7509              <1>       jne    short sysrename_5 ; Bad file name
19258                            <1>
19259                            <1>       ; no path name !? (rename file in same directory)
19260 0001285B 803D[1D940100]20  <1>       cmp    byte [DestinationFile_Directory], 20h
19261 00012862 7607              <1>       jna    short sysrename_6
19262                            <1> sysrename_5:
19263 00012864 B801000000        <1>       mov    eax, ERR_BAD_CMD_ARG ; 1 = Bad file name
19264                            <1>                                   ; (Bad argument)
19265 00012869 EB82              <1>       jmp    short sysrename_error
19266                            <1> sysrename_6:
19267 0001286B 803D[5E940100]20  <1>       cmp    byte [DestinationFile_Name], 20h
19268 00012872 76F0              <1>       jna    short sysrename_5
19269                            <1>
19270 00012874 BE[5E940100]      <1>       mov    esi, DestinationFile_Name
19271 00012879 E8916EFFFF        <1>       call   check_filename ; is it a valid msdos file name?
19272 0001287E 0F8269FFFFFF      <1>       jc     sysrename_error ; 26 = ERR_INV_FILE_NAME
19273                            <1>
19274                            <1>       ;mov   esi, DestinationFile_Name
19275 00012884 66B80008          <1>       mov    ax, 0800h ; Find File or Directory
19276 00012888 E8C46AFFFF        <1>       call   find_first_file
19277 0001288D 720A              <1>       jc     short sysrename_7
19278                            <1>
19279 0001288F B80E000000        <1>       mov    eax, ERR_FILE_EXISTS  ; file already exists !
19280 00012894 E954FFFFFF        <1>       jmp    sysrename_error
19281                            <1> sysrename_7:
19282                            <1>       ; eax = 2 (File not found !)
19283 00012899 3C02              <1>       cmp    al, 2 ; ERR_NOT_FOUND
19284 0001289B 0F854CFFFFFF      <1>       jne    sysrename_error
19285                            <1>
19286                            <1>       ; 31/12/2017
19287                            <1>       ; Following code is also part of 'rename_file' in
19288                            <1>       ; 'trdosk3.s' (MainProg's 'rename' command) ; 13/11/2017
19289 000128A1 BE[5E940100]      <1>       mov    esi, DestinationFile_Name ; (Rename_NewName)
```

```
19290 000128A6 668B0D[16940100]    <1>       mov    cx, [SourceFile_DirEntryNumber]
19291 000128AD 66A1[02940100]      <1>       mov    ax, [SourceFile_DirEntry+20] ; First Cluster, HW
19292 000128B3 C1E010              <1>       shl    eax, 16
19293 000128B6 66A1[08940100]      <1>       mov    ax, [SourceFile_DirEntry+26] ; First Cluster, LW
19294 000128BC 0FB61D[EB930100]    <1>       movzx  ebx, byte [SourceFile_LongNameEntryLength]
19295 000128C3 E8A294FFFF          <1>       call   rename_directory_entry
19296 000128C8 0F821FFFFFFF        <1>       jc     sysrename_error
19297                              <1>       ;xor   eax, eax
19298 000128CE A3[64030300]        <1>       mov    [u.r0], eax ; 0
19299                              <1>       ;mov   [u.error], eax
19300 000128D3 E89D060000          <1>       call   reset_working_path
19301 000128D8 E9EFAEFFFF          <1>       jmp    sysret
19302                              <1>
19303                              <1> sysmem: ; Get Total&Free Memory amount
19304                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19305                              <1>       ;
19306                              <1>         ; INPUT ->
19307                              <1>       ;      none
19308                              <1>       ; OUTPUT ->
19309                              <1>       ;      EAX = Total memory count (in bytes)
19310                              <1>       ;      EBX = Virtually available memory amount (in bytes)
19311                              <1>       ;            = 4GB - CORE (4MB)
19312                              <1>       ;      ECX = Free memory count (in bytes)
19313                              <1>       ;      EDX = Calculated free memory count (in bytes)
19314                              <1>
19315 000128DD A1[F4880100]        <1>       mov    eax, [memory_size] ; in pages
19316 000128E2 C1E00C              <1>       shl    eax, 12                ; in bytes
19317 000128E5 A3[64030300]        <1>       mov    [u.r0], eax
19318 000128EA E8C119FFFF          <1>       call   calc_free_mem
19319                              <1>       ; edx = calculated free pages
19320                              <1>       ; ecx = 0
19321 000128EF 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; EBP points to user's registers
19322 000128F5 C745100000C0FF      <1>       mov    dword [ebp+16], ECORE ; EBX (for user)
19323                              <1>                              ; 0FFC00000h ; 4GB - 4MB
19324 000128FC C1E20C              <1>       shl    edx, 12
19325 000128FF 895514              <1>       mov    [ebp+20], edx ; EDX (for user)
19326 00012902 8B0D[F8880100]      <1>       mov    ecx, [free_pages]
19327 00012908 C1E10C              <1>       shl    ecx, 12       ; free bytes
19328 0001290B 894D18              <1>       mov    [ebp+24], ecx ; ECX (for user)
19329                              <1>       ;mov   [free_pages], edx
19330 0001290E E9B9AEFFFF          <1>       jmp    sysret
19331                              <1>
19332                              <1> sysprompt:
19333                              <1>       ; Set TRDOS 386 Command Interpreter (MainProg) prompt
19334                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19335                              <1>       ;
19336                              <1>         ; INPUT ->
19337                              <1>       ;      EBX = 0 -> use default prompt
19338                              <1>       ;      EBX > 0 -> prompt string (ASCIIZ) address
19339                              <1>       ;                 (Max. 11 characters except ZERO tail)
19340                              <1>       ; OUTPUT ->
19341                              <1>       ;      (EAX = 0)
19342                              <1>       ;      CF = 0 -> Successful
19343                              <1>       ;      CF = 1 -> Failed
19344                              <1>
19345 00012913 21DB                <1>       and    ebx, ebx
19346 00012915 750A                <1>       jnz    short sysprompt_0
19347                              <1>
19348 00012917 E83964FFFF          <1>       call   default_command_prompt ; '['+'TRDOS'+']'
19349 0001291C E9ABAEFFFF          <1>       jmp    sysret
19350                              <1>
19351                              <1> sysprompt_0:
19352 00012921 31C0                <1>       xor    eax, eax
19353 00012923 A3[64030300]        <1>       mov    [u.r0], eax
19354 00012928 89DE                <1>       mov    esi, ebx
19355 0001292A B90C000000          <1>       mov    ecx, 12
19356 0001292F 89E5                <1>       mov    ebp, esp
19357 00012931 29CC                <1>       sub    esp, ecx
19358 00012933 49                  <1>       dec    ecx ; 11
19359 00012934 89E7                <1>       mov    edi, esp
19360 00012936 E833F1FFFF          <1>       call   transfer_from_user_buffer
19361 0001293B 7211                <1>       jc     short sysprompt_err
19362 0001293D 803E20              <1>       cmp    byte [esi], 20h
19363 00012940 760C                <1>       jna    short sysprompt_err
19364 00012942 E82064FFFF          <1>       call   set_command_prompt
19365 00012947 89EC                <1>       mov    esp, ebp
19366 00012949 E97EAEFFFF          <1>       jmp    sysret
19367                              <1> sysprompt_err:
19368                              <1> syspath_err:
19369 0001294E 89EC                <1>       mov    esp, ebp
19370 00012950 E957AEFFFF          <1>       jmp    error
19371                              <1>
19372                              <1> syspath:
19373                              <1>       ; Get/Set Run Path
19374                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19375                              <1>       ;
19376                              <1>         ; INPUT ->
19377                              <1>       ;      EBX = 0 -> get path (to buffer address in ECX)
19378                              <1>       ;      EBX > 0 -> set path
19379                              <1>       ;          EBX = Path string buffer address (ASCIIZ)
19380                              <1>       ;               (Path description except 'PATH=')
19381                              <1>       ;      ECX = Buffer address (if EBX = 0)
19382                              <1>       ;            (ECX will not be used if EBX > 0)
19383                              <1>       ;      DL = Buffer size (0 = 256 byte)
19384                              <1>       ;
19385                              <1>       ; OUTPUT ->
19386                              <1>       ;      CF = 0 -> Successful (EAX = String length)
19387                              <1>       ;      CF = 1 -> Failed (EAX = 0)
19388                              <1>       ;
19389                              <1>       ; NOTE: 'PATH=' or 'PATH' must be excluded
19390                              <1>       ;  (It must not be at the beginning of the string.)
19391                              <1>
19392 00012955 89E5                <1>       mov    ebp, esp
19393 00012957 81EC00010000        <1>       sub    esp, 256
19394 0001295D 89E7                <1>       mov    edi, esp
```

```
19395                                  <1>
19396 0001295F 31C0                    <1>       xor    eax, eax
19397 00012961 A3[64030300]            <1>       mov    [u.r0], eax
19398                                  <1>
19399 00012966 21DB                    <1>       and    ebx, ebx
19400 00012968 752E                    <1>       jnz    short syspath_0
19401                                  <1>
19402                                  <1>       ; EBX = 0 -> get run path
19403 0001296A 89CB                    <1>       mov    ebx, ecx ; buffer addr (in user's mem space)
19404 0001296C BE[4F400100]            <1>       mov    esi, Cmd_Path  ; 'PATH' address
19405 00012971 0FB6CA                  <1>       movzx  ecx, dl
19406 00012974 80E901                  <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
19407 00012977 6683D101                <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
19408                                  <1>       ; EDI = Output buffer
19409                                  <1>       ; CX = Buffer length
19410                                  <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
19411                                  <1>       ; ESI = 'PATH' address (with zero tail)
19412 0001297B E81B7CFFFF              <1>       call   get_environment_string
19413 00012980 72CC                    <1>       jc     short syspath_err
19414 00012982 89DF                    <1>       mov    edi, ebx ; User's buffer address
19415 00012984 89E6                    <1>       mov    esi, esp
19416                                  <1>       ; EDI = User's buffer address
19417                                  <1>       ; ECX = transfer (byte) count
19418 00012986 E899F0FFFF              <1>       call   transfer_to_user_buffer
19419 0001298B 72C1                    <1>       jc     short syspath_err
19420 0001298D 890D[64030300]          <1>       mov    [u.r0], ecx
19421 00012993 E934AEFFFF              <1>       jmp    sysret
19422                                  <1>
19423                                  <1> syspath_0:
19424 00012998 89DE                    <1>       mov    esi, ebx
19425 0001299A 0FB6CA                  <1>       movzx  ecx, dl
19426 0001299D 80E901                  <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
19427 000129A0 6683D101                <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
19428 000129A4 E8C5F0FFFF              <1>       call   transfer_from_user_buffer
19429 000129A9 72A3                    <1>       jc     short syspath_err
19430                                  <1>       ;(*) 'PATH=' will be added to
19431                                  <1>       ;         the head of the string
19432 000129AB 83EC08                  <1>       sub    esp, 8 ;(*)
19433 000129AE 89FE                    <1>       mov    esi, edi ;(*)
19434 000129B0 E8CA7BFFFF              <1>       call   set_path_x ;(*)
19435 000129B5 7297                    <1>       jc     short syspath_err
19436 000129B7 8915[64030300]          <1>       mov    [u.r0], edx ; run path string length
19437 000129BD E90AAEFFFF              <1>       jmp    sysret
19438                                  <1>
19439                                  <1> sysenv:
19440                                  <1>       ; Get/Set Environment Variables
19441                                  <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19442                                  <1>       ;
19443                                  <1>         ; INPUT ->
19444                                  <1>       ;      EBX = 0 -> get (all) environment variables
19445                                  <1>       ;            (Required Buffer length = 512 bytes)
19446                                  <1>       ;      EBX > 0 -> set (one) environment variable
19447                                  <1>       ;            (If there is not a '=' after
19448                                  <1>       ;            the environment variable name, it will
19449                                  <1>       ;            accepted as 'get environment variable'.)
19450                                  <1>       ;             EBX = Buffer address
19451                                  <1>       ;      ECX = Buffer address (if EBX = 0)
19452                                  <1>       ;            (ECX will not be used if EBX > 0)
19453                                  <1>       ;            (Note: Buffer size is 512 bytes.)
19454                                  <1>       ;      DL = Buffer size (0 = 256 byte)
19455                                  <1>       ;           (For one envrionment variable)
19456                                  <1>       ;
19457                                  <1>       ; OUTPUT ->
19458                                  <1>       ;      (EAX = 0)
19459                                  <1>       ;      CF = 0 -> Successful (EAX = String length)
19460                                  <1>       ;      CF = 1 -> Failed (EAX = 0)
19461                                  <1>       ;
19462                                  <1>       ; Note: Environment variable name, for example,
19463                                  <1>       ;      'PATH=' must be included at the beginning
19464                                  <1>       ;      of the environment string. If the variable
19465                                  <1>       ;      name is as 'PATH' but it is not as 'PATH='
19466                                  <1>       ;      the variable string (row) will be returned.
19467                                  <1>       ;      If variable name is as 'PATH=' but there is
19468                                  <1>       ;      not a following text after the variable name,
19469                                  <1>       ;      the environment variable will be reset/deleted.
19470                                  <1>
19471 000129C2 89E5                    <1>       mov    ebp, esp
19472 000129C4 81EC00020000            <1>       sub    esp, 512
19473 000129CA 89E7                    <1>       mov    edi, esp
19474                                  <1>
19475 000129CC 31C0                    <1>       xor    eax, eax
19476 000129CE A3[64030300]            <1>       mov    [u.r0], eax
19477                                  <1>
19478 000129D3 21DB                    <1>       and    ebx, ebx
19479 000129D5 7524                    <1>       jnz    short sysenv_0
19480                                  <1>
19481                                  <1>       ; EBX = 0 -> get (all) environment variables
19482 000129D7 89EC                    <1>       mov    esp, ebp
19483 000129D9 BE00300900              <1>       mov    esi, Env_Page  ; Environment page
19484 000129DE 89CF                    <1>       mov    edi, ecx ; buffer addr (in user's mem space)
19485 000129E0 B900020000              <1>       mov    ecx, 512
19486 000129E5 E83AF0FFFF              <1>       call   transfer_to_user_buffer
19487 000129EA 0F82BCADFFFF            <1>       jc     error
19488 000129F0 890D[64030300]          <1>       mov    [u.r0], ecx
19489 000129F6 E9D1ADFFFF              <1>       jmp    sysret
19490                                  <1>
19491                                  <1> sysenv_0:
19492 000129FB 89DE                    <1>       mov    esi, ebx ; * ; user's buffer address
19493 000129FD 0FB6CA                  <1>       movzx  ecx, dl
19494 00012A00 80E901                  <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
19495 00012A03 6683D101                <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
19496 00012A07 E862F0FFFF              <1>       call   transfer_from_user_buffer
19497 00012A0C 723F                    <1>       jc     short sysenv_err
19498 00012A0E 89FE                    <1>       mov    esi, edi
19499 00012A10 8A06                    <1>       mov    al, [esi]
```

```
19500 00012A12 3C20              <1>       cmp    al, 20h
19501 00012A14 7637              <1>       jna    short sysenv_err
19502 00012A16 3C3D              <1>       cmp    al, '='
19503 00012A18 7433              <1>       je     short sysenv_err
19504 00012A1A 56                <1>       push   esi
19505                            <1> sysenv_1:
19506 00012A1B 46                <1>       inc    esi
19507 00012A1C 803E3D            <1>       cmp    byte [esi], '='
19508 00012A1F 7433              <1>       je     short sysenv_3
19509 00012A21 803E20            <1>       cmp    byte [esi], 20h
19510 00012A24 73F5              <1>       jnb    short sysenv_1
19511 00012A26 C60600            <1>       mov    byte [esi], 0
19512 00012A29 5E                <1>       pop    esi
19513                            <1>       ; EDI = Output buffer
19514                            <1>       ; CX = Buffer length
19515 00012A2A 30C0              <1>       xor    al, al
19516                            <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
19517                            <1>       ; ESI = Environment variable name address
19518 00012A2C E86A7BFFFF        <1>       call   get_environment_string
19519 00012A31 721A              <1>       jc     short sysenv_err
19520 00012A33 89DF              <1>       mov    edi, ebx ; * ; user's buffer address
19521 00012A35 89C1              <1>       mov    ecx, eax ; String length
19522 00012A37 89E6              <1>       mov    esi, esp
19523                            <1>       ; ESI = system buffer address
19524                            <1>       ; EDI = User's buffer address
19525                            <1>       ; ECX = transfer (byte) count
19526 00012A39 E8E6EFFFFF        <1>       call   transfer_to_user_buffer
19527 00012A3E 720D              <1>       jc     short sysenv_err
19528 00012A40 890D[64030300]    <1>       mov    [u.r0], ecx ; transfer (byte) count
19529                            <1> sysenv_2:
19530 00012A46 89EC              <1>       mov    esp, ebp
19531 00012A48 E97FADFFFF        <1>       jmp    sysret
19532                            <1> sysenv_err:
19533 00012A4D 89EC              <1>       mov    esp, ebp
19534 00012A4F E958ADFFFF        <1>       jmp    error
19535                            <1> sysenv_3:
19536 00012A54 46                <1>       inc    esi
19537 00012A55 803E20            <1>       cmp    byte [esi], 20h
19538 00012A58 73FA              <1>       jnb    short sysenv_3
19539 00012A5A C60600            <1>       mov    byte [esi], 0
19540 00012A5D 5E                <1>       pop    esi
19541 00012A5E E8FB7BFFFF        <1>       call   set_environment_string
19542 00012A63 72E8              <1>       jc     short sysenv_err
19543 00012A65 8915[64030300]    <1>       mov    [u.r0], edx
19544 00012A6B EBD9              <1>       jmp    short sysenv_2
19545                            <1>
19546                            <1>
19547                            <1> ; 22/01/2021
19548                            <1> ; temporary - 24/01/2016
19549                            <1>
19550                            <1> iget:
19551                            <1>       ;retn
19552                            <1> isintr:
19553                            <1>       ;retn
19554                            <1> iopen:
19555                            <1>       ;retn
19556                            <1> iclose:
19557                            <1>       ;retn
19558                            <1> sndc:
19559                            <1>       ;retn
19560                            <1> access:
19561                            <1>       ;retn
19562                            <1> sleep:
19563 00012A6D C3                <1>       retn
 3090                               %include 'trdosk7.s' ; 24/01/2016
 3091                            <1> ; ****************************************************************************
 3092                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DISK READ&WRITE : trdosk7.s
 3093                            <1> ; ----------------------------------------------------------------------------
 3094                            <1> ; Last Update: 25/02/2016
 3095                            <1> ; ----------------------------------------------------------------------------
 3096                            <1> ; Beginning: 24/01/2016
 3097                            <1> ; ----------------------------------------------------------------------------
 3098                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
 3099                            <1> ; ----------------------------------------------------------------------------
 3100                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
 3101                            <1> ; DISK_IO.ASM (20/07/2011)
 3102                            <1> ; ****************************************************************************
 3103                            <1> ; DISK_IO.ASM (c) 2009-2011 Erdogan TAN [ 04/07/2009 ] Last Update: 20/07/2011
 3104                            <1>
 3105                            <1> disk_write:
 3106                            <1>       ; 25/02/2016
 3107                            <1>       ; 24/02/2016
 3108                            <1>       ; 23/02/2016
 3109 00012A6E 807E0500          <1>       cmp    byte [esi+LD_LBAYes], 0
 3110 00012A72 777B              <1>       ja     short lba_write
 3111                            <1>
 3112                            <1> chs_write:
 3113                            <1>       ; 25/02/2016
 3114                            <1>       ; 23/02/2016
 3115 00012A74 C605[A5910100]03  <1>       mov    byte [disk_rw_op], 3 ; CHS write
 3116 00012A7B EB0D              <1>       jmp    short chs_rw
 3117                            <1>
 3118                            <1> disk_read:
 3119                            <1>       ; 25/02/2016
 3120                            <1>       ; 24/02/2016
 3121                            <1>       ; 23/02/2016
 3122                            <1>       ; 17/02/2016
 3123                            <1>       ; 14/02/2016
 3124                            <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
 3125                            <1>       ; 17/10/2010
 3126                            <1>       ; 18/04/2010
 3127                            <1>       ;
 3128                            <1>       ; INPUT -> EAX = Logical Block Address
 3129                            <1>       ;          ESI = Logical Dos Disk Table Offset (DRV)
 3130                            <1>       ;          ECX = Sector Count
```

```
3131                                    <1>       ;         EBX = Destination Buffer
3132                                    <1>       ; OUTPUT ->
3133                                    <1>       ;         cf = 0 or cf = 1
3134                                    <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
3135                                    <1>
3136 00012A7D 807E0500                 <1>       cmp    byte [esi+LD_LBAYes], 0
3137 00012A81 7775                     <1>         ja      short lba_read
3138                                    <1>
3139                                    <1> chs_read:
3140                                    <1>       ; 25/02/2016
3141                                    <1>       ; 24/02/2016
3142                                    <1>       ; 23/02/2016
3143                                    <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
3144                                    <1>       ; 20/07/2011
3145                                    <1>       ; 04/07/2009
3146                                    <1>       ;
3147                                    <1>       ; INPUT -> EAX = Logical Block Address
3148                                    <1>       ;         ECX = Number of sectors to read
3149                                    <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
3150                                    <1>       ;         EBX = Destination Buffer
3151                                    <1>       ; OUTPUT ->
3152                                    <1>       ;         cf = 0 or cf = 1
3153                                    <1>       ; (Modified registers: EAX; EBX, ECX, EDX)
3154                                    <1>
3155                                    <1>       ; 23/02/2016
3156 00012A83 C605[A5910100]02         <1>       mov    byte [disk_rw_op], 2 ; CHS read
3157                                    <1>
3158                                    <1> chs_rw:
3159                                    <1>       ;;movzx      edx, word [esi+LD_BPB+SecPerTrack]
3160                                    <1>       ;movzx edx, byte [esi+LD_BPB+SecPerTrack] ; <= 63
3161                                    <1>       ;mov   [disk_rw_spt], dl
3162                                    <1>
3163                                    <1> chs_read_next_sector:
3164 00012A8A C605[A6910100]04         <1>       mov    byte [retry_count], 4
3165                                    <1>
3166                                    <1> chs_read_retry:
3167                                    <1>       ;mov   [sector_count], ecx ; 23/02/2016
3168                                    <1>
3169 00012A91 50                       <1>       push   eax                ; Linear sector #
3170 00012A92 51                       <1>       push   ecx                ; # of FAT/FILE/DIR sectors
3171                                    <1>
3172 00012A93 0FB74E1E                 <1>       movzx  ecx, word [esi+LD_BPB+SecPerTrack]
3173                                    <1>       ;movzx ecx, byte [disk_rw_spt] ; 23/02/2016
3174 00012A97 29D2                     <1>       sub    edx, edx
3175 00012A99 F7F1                     <1>       div    ecx
3176                                    <1>       ; eax = track, dx (dl ) = sector (on track)
3177                                    <1>       ;sub   cl, dl ; 24/02/2016 (spt - sec)
3178                                    <1>       ;push  ecx ; *
3179 00012A9B 6689D1                   <1>       mov    cx, dx             ; Sector (zero based)
3180 00012A9E 6641                     <1>       inc    cx                 ; To make it 1 based
3181 00012AA0 6651                     <1>       push   cx
3182 00012AA2 668B4E20                 <1>       mov    cx, [esi+LD_BPB+Heads]
3183 00012AA6 6629D2                   <1>       sub    dx, dx
3184 00012AA9 F7F1                     <1>       div    ecx                ; Convert track to head & cyl
3185                                    <1>       ; eax (ax) = cylinder, dx (dl) = head (max. FFh)
3186 00012AAB 88D6                     <1>       mov    dh, dl
3187 00012AAD 6659                     <1>       pop    cx                 ; AX=Cyl, DH=Head, CX=Sector
3188 00012AAF 8A5602                   <1>       mov    dl, [esi+LD_PhyDrvNo]
3189                                    <1>
3190 00012AB2 88C5                     <1>       mov    ch, al             ; NOTE: max. 1023 cylinders !
3191 00012AB4 C0CC02                   <1>       ror    ah, 2              ; Rotate 2 bits right
3192 00012AB7 08E1                     <1>       or     cl, ah
3193                                    <1>
3194                                    <1>       ; 24/02/2016
3195                                    <1>       ;pop   eax ; * (spt - sec) (example: 63 - 0 = 63)
3196                                    <1>       ;cmp   eax, [sector_count]
3197                                    <1>       ;jb    short chs_write_sectors
3198                                    <1>       ;je    short chs_read_sectors
3199                                    <1>       ;; (# of sectors to read is more than remaining sectors on the track)
3200                                    <1>       ;mov   al, [sector_count]
3201                                    <1> ;chs_read_sectors: ; read or write !
3202 00012AB9 B001                     <1>       mov    al, 1 ; 25/02/2016
3203 00012ABB 8A25[A5910100]           <1>       mov    ah, [disk_rw_op]  ; 02h = chs read, 03h = chs write
3204                                    <1>       ;
3205 00012AC1 E8C327FFFF               <1>       call   int13h             ; BIOS Service func ( ah ) = 2
3206                                    <1>                                 ; Read disk sectors
3207                                    <1>                                 ; AL-sec num CH-track CL-sec
3208                                    <1>                                 ; DH-head DL-drive ES:BX-buffer
3209                                    <1>                                 ; CF-flag AH-stat AL-sec read
3210                                    <1>                                ; If CF = 1 then (If AH > 0)
3211 00012AC6 8825[A7910100]           <1>       mov    [disk_rw_err], ah
3212                                    <1>
3213 00012ACC 59                       <1>       pop    ecx
3214 00012ACD 58                       <1>       pop    eax
3215 00012ACE 7314                     <1>       jnc    short chs_read_ok
3216                                    <1>
3217 00012AD0 803D[A7910100]09         <1>       cmp    byte [disk_rw_err], 09h ; DMA crossed 64K segment boundary
3218 00012AD7 7408                     <1>       je     short chs_read_error_retn
3219                                    <1>
3220 00012AD9 FE0D[A6910100]           <1>       dec    byte [retry_count]
3221 00012ADF 75B0                     <1>       jnz    short chs_read_retry
3222                                    <1>
3223                                    <1> chs_read_error_retn:
3224 00012AE1 F9                       <1>       stc
3225                                    <1>       ;retn
3226 00012AE2 EB69                     <1>       jmp    short update_drv_error_byte
3227                                    <1>
3228                                    <1> ;chs_write_sectors: ; read or write
3229                                    <1>       ;; (# of sectors to read is less than remaining sectors on the track)
3230                                    <1>       ;mov   [sector_count], al
3231                                    <1>       ;jmp   short chs_read_sectors
3232                                    <1>
3233                                    <1> chs_read_ok:
3234                                    <1>       ;; 23/02/2016
3235                                    <1>       ;movzx edx, byte [sector_count] ; sector count (<= spt)
```

```
                                <1>         ;sub    ecx, edx  ; remaining sector count
                                <1>       ;jna   short update_drv_error_byte
                                <1>       ;add   eax, edx ; next disk sector
                                <1>       ;shl   edx, 9 ; 512 * sector count
                                <1>       ;add   ebx, edx ; next buffer byte address
                                <1>         ;jmp     chs_read_next_sector
                                <1>       ; 25/02/2016
00012AE4 40                     <1>       inc    eax ; next sector
00012AE5 81C300020000           <1>       add    ebx, 512
00012AEB E29D                   <1>       loop   chs_read_next_sector
00012AED EB5E                   <1>       jmp    short update_drv_error_byte
                                <1>
                                <1> lba_write:
                                <1>       ; 23/02/2016
00012AEF C605[A5910100]1C       <1>       mov    byte [disk_rw_op], 1Ch ; LBA write
00012AF6 EB07                   <1>       jmp    short lba_rw
                                <1>
                                <1> lba_read:
                                <1>       ; 23/02/2016
                                <1>       ; 17/02/2016
                                <1>       ; 14/02/2016
                                <1>       ; 13/02/2016
                                <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 10/07/2015 (Retro UNIX 386 v1)
                                <1>       ;
                                <1>       ; INPUT -> EAX = Logical Block Address
                                <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
                                <1>       ;         ECX = Sector Count
                                <1>       ;         EBX = Destination Buffer
                                <1>       ; OUTPUT ->
                                <1>       ;         cf = 0 or cf = 1
                                <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
                                <1>
                                <1>       ; LBA read/write (with private LBA function)
                                <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
                                <1>
                                <1>
                                <1>       ; 23/02/2016
00012AF8 C605[A5910100]1B       <1>       mov    byte [disk_rw_op], 1Bh ; LBA read
                                <1>
                                <1> lba_rw:
                                <1>       ; 17/02/2016
00012AFF 57                     <1>       push   edi
                                <1>
00012B00 890D[A8910100]         <1>       mov    [sector_count], ecx ; total sector (read) count
                                <1>
00012B06 8A5602                 <1>       mov    dl, [esi+LD_PhyDrvNo]
                                <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
                                <1>
                                <1> lba_read_next:
00012B09 81F900010000           <1>       cmp    ecx, 256
00012B0F 7605                   <1>       jna    short lba_read_rsc
00012B11 B900010000             <1>       mov    ecx, 256 ; 17/02/2016
                                <1> lba_read_rsc:
00012B16 290D[A8910100]         <1>       sub    [sector_count], ecx ; remain sectors
                                <1>
00012B1C 89CF                   <1>       mov    edi, ecx
00012B1E 89C1                   <1>       mov    ecx, eax ; sector number/address
                                <1>
00012B20 C605[A6910100]04       <1>       mov    byte [retry_count], 4
                                <1> lba_read_retry:
00012B27 89F8                   <1>       mov    eax, edi
                                <1>       ;
                                <1>       ; ecx = sector number
                                <1>       ; al = sector count (0 - 255) /// (0 = 256)
                                <1>       ; dl = drive number
                                <1>       ; ebx = buffer offset
                                <1>       ;
                                <1>       ; Function 1Bh = LBA read, 1Ch = LBA write
                                <1>       ; 23/02/2016
00012B29 8A25[A5910100]         <1>       mov    ah, [disk_rw_op] ; 1Bh = LBA read, 1Ch = LBA write
00012B2F E85527FFFF             <1>       call   int13h
                                <1>       ; al = ? (changed)
                                <1>       ; ah = error code
00012B34 8825[A7910100]         <1>       mov    [disk_rw_err], ah
00012B3A 7334                   <1>       jnc    short lba_read_ok
00012B3C 80FC80                 <1>       cmp    ah, 80h ; time out?
00012B3F 740A                   <1>         je     short lba_read_stc_retn
00012B41 FE0D[A6910100]         <1>       dec    byte [retry_count]
00012B47 7FDE                   <1>       jg     short lba_read_retry
00012B49 743A                   <1>       jz     short lba_read_reset
                                <1>       ; sf =  1
                                <1>
                                <1> lba_read_stc_retn:
00012B4B F9                     <1>       stc
                                <1> lba_read_retn:
00012B4C 5F                     <1>       pop    edi
                                <1>
                                <1> update_drv_error_byte:
00012B4D 9C                     <1>       pushf
00012B4E 53                     <1>       push   ebx
00012B4F 6651                   <1>       push   cx
                                <1>       ;or    ecx, ecx
                                <1>       ;jz    short udrv_errb0
00012B51 8A0D[A7910100]         <1>       mov    cl, [disk_rw_err]
                                <1> udrv_errb0:
00012B57 0FB65E02               <1>       movzx  ebx, byte [esi+LD_PhyDrvNo]
00012B5B 80FB02                 <1>       cmp    bl, 2
00012B5E 7203                   <1>         jb     short udrv_errb1
00012B60 80EB7E                 <1>       sub    bl, 7Eh
                                <1>       ;cmp   bl, 5
                                <1>       ;ja    short udrv_errb2
                                <1> udrv_errb1:
00012B63 81C3[216D0000]         <1>         add     ebx, drv.error ; 13/02/2016
00012B69 880B                   <1>       mov    [ebx], cl ; error code
```

```
3341                            <1> udrv_errb2:
3342 00012B6B 6659              <1>       pop    cx
3343 00012B6D 5B                <1>       pop    ebx
3344 00012B6E 9D                <1>       popf
3345 00012B6F C3                <1>       retn
3346                            <1>
3347                            <1> lba_read_ok:
3348 00012B70 89C8              <1>       mov    eax, ecx ; sector number
3349 00012B72 01F8              <1>       add    eax, edi ; sector number (next)
3350 00012B74 C1E709            <1>       shl    edi, 9 ; sector count * 512
3351 00012B77 01FB              <1>       add    ebx, edi ; next buffer offset
3352                            <1>
3353 00012B79 8B0D[A8910100]    <1>       mov    ecx, [sector_count] ; remaining sectors
3354 00012B7F 09C9              <1>       or     ecx, ecx
3355 00012B81 7586              <1>       jnz    short lba_read_next
3356 00012B83 EBC7              <1>       jmp    short lba_read_retn
3357                            <1>
3358                            <1> lba_read_reset:
3359 00012B85 B40D              <1>       mov    ah, 0Dh ; Alternate reset
3360 00012B87 E8FD26FFFF        <1>       call   int13h
3361                            <1>       ; al = ? (changed)
3362                            <1>       ; ah = error code
3363 00012B8C 7399              <1>       jnc    short lba_read_retry
3364 00012B8E EBBC              <1>       jmp    short lba_read_retn
3091                                %include 'trdosk8.s' ; 24/01/2016
3092                            <1> ; ****************************************************************************
3093                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.2) - MAIN PROGRAM : trdosk8.s
3094                            <1> ; ----------------------------------------------------------------------------
3095                            <1> ; Last Update: 12/02/2021
3096                            <1> ; ----------------------------------------------------------------------------
3097                            <1> ; Beginning: 24/01/2016
3098                            <1> ; ----------------------------------------------------------------------------
3099                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3100                            <1> ; ----------------------------------------------------------------------------
3101                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3102                            <1> ; u0.s (20/11/2015), u4.s (14/10/2015)
3103                            <1> ; ****************************************************************************
3104                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3105                            <1> ; TRDOS2.ASM (09/11/2011)
3106                            <1> ; ----------------------------------------------------------------------------
3107                            <1> ; DIR.ASM (c) 2004-2011 Erdogan TAN  [07/01/2004] Last Update: 09/10/2011
3108                            <1>
3109                            <1> set_run_sequence:
3110                            <1>       ; 23/12/2016
3111                            <1>       ; 10/06/2016
3112                            <1>       ; 22/05/2016
3113                            <1>       ; 20/05/2016
3114                            <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
3115                            <1>       ; TRDOS 386 feature only !
3116                            <1>       ;
3117                            <1>       ; INPUT ->
3118                            <1>       ;     AL = process number (next process)
3119                            <1>       ;
3120                            <1>       ;     this process must be added to run sequence
3121                            <1>       ;
3122                            <1>       ;     [u.pri] = priority of present process
3123                            <1>       ;
3124                            <1>       ;     DL = priority (queue)
3125                            <1>       ;          0 = background (low) ; run on background
3126                            <1>       ;          1 = regular (normal) ; run as regular
3127                            <1>       ;          2 = event (high)     ; run for event
3128                            <1>       ;
3129                            <1>       ;     1) If the requested process is already running:
3130                            <1>       ;        a) If present priority is high ([u.pri]=2)
3131                            <1>       ;           and requested priority is also high,
3132                            <1>       ;           there is nothing to do! Because it has been
3133                            <1>       ;           done already (before this attempt).
3134                            <1>       ;        b) If present priority is high ([u.pri]=2)
3135                            <1>       ;           and requested priority is not high, there is
3136                            <1>       ;           nothing to do! Because, it's current
3137                            <1>       ;           run queue is unspecified, here. (It may be in
3138                            <1>       ;           a waiting list or in a run queue; if the new
3139                            <1>       ;           priority would be used to add it to relavant
3140                            <1>       ;           run queue, this would be wrong, unnecessary
3141                            <1>       ;           and destabilizing duplication!)
3142                            <1>       ;        c) If present priority is not high ([u.pri]<2)
3143                            <1>       ;           and requested priority is high (event),
3144                            <1>       ;           process will be added to present priority's
3145                            <1>       ;           run queue and then, priority will be changed
3146                            <1>       ;           to high ([u.pri]=2).
3147                            <1>       ;        d) If present priority is not high ([u.pri]<2)
3148                            <1>       ;           and requested priority is not high, [u.pri]
3149                            <1>       ;           value will be changed. There is nothing to do
3150                            <1>       ;           in addition. (The new priority value will be
3151                            <1>       ;           used by 'tswap/tswitch' procedure at 'sysret'
3152                            <1>       ;           or 'sysrele' stage.)
3153                            <1>       ;
3154                            <1>       ;     2) If the requested process is not running:
3155                            <1>       ;        a) If requested priority of the requested
3156                            <1>       ;           (next) process is high (event) and priority
3157                            <1>       ;           of present process is not high, the requested
3158                            <1>       ;           process will be added to ('runq_event') high
3159                            <1>       ;           priority run queue and then present (running)
3160                            <1>       ;           process will be stopped (swapped/switched out)
3161                            <1>       ;           immediately if it is in user mode, or it's
3162                            <1>       ;           [u.quant] value will be reset to 0 and (then)
3163                            <1>       ;           it will be stopped at 'sysret' stage.
3164                            <1>       ;        b) If requested priority of the requested
3165                            <1>       ;           (next) process is high (event) and priority
3166                            <1>       ;           of present process is also high, the requested
3167                            <1>       ;           process will be added to ('runq_event') high
3168                            <1>       ;           priority run queue and present (running)
3169                            <1>       ;           process will be allowed to run until it's
3170                            <1>       ;           time quantum will be elapsed ([u.quant]=0).
3171                            <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:
00012B90 3A05[B3030300]         <1>       cmp   al, [u.uno]     ; same process ?
00012B96 750C                   <1>       jne   short srunseq_2 ; no
                                <1>
00012B98 8A25[A9030300]         <1>       mov   ah, [u.pri]  ; present/current priority
00012B9E 80FC02                 <1>       cmp   ah, 2        ; 'run for event' priority level
00012BA1 7221                   <1>       jb    short srunseq_6 ; no
                                <1>
                                <1> srunseq_1:
                                <1>       ; there is nothing to do!
00012BA3 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
00012BA4 80FA02                 <1>       cmp   dl, 2        ; event queue
00012BA7 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
00012BA9 3A15[A9030300]         <1>       cmp   dl, [u.pri]  ; priority of present process
00012BAF 7606                   <1>       jna   short srunseq_3 ; is high, also
                                <1>       ;
                                <1>       ; present process will be swapped/switched out
00012BB1 FE05[81950100]         <1>       inc   byte [p_change] ; 1
                                <1>
                                <1> srunseq_3:
                                <1>       ; add process to 'runq_event' queue for new event
00012BB7 BB[52030300]           <1>       mov   ebx, runq_event ; high priority run queue
                                <1>
                                <1> srunseq_4:
                                <1>       ; al = process number
                                <1>       ; ebx = run queue
00012BBC E887EDFFFF             <1>       call  putlu
00012BC1 C3                     <1>       retn
                                <1>
                                <1> srunseq_5:
00012BC2 F5                     <1>       cmc
00012BC3 C3                     <1>       retn
                                <1>
                                <1> srunseq_6:
                                <1>       ; present priority of the process is not high
                                <1>
00012BC4 8815[A9030300]         <1>       mov   [u.pri], dl ; new priority
                                <1>                         ; (will be used by 'tswap')
                                <1>
00012BCA 80FA02                 <1>       cmp   dl, 2        ; high priority ?
00012BCD 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>
00012BCF BB[54030300]           <1>       mov   ebx, runq_normal ; 'run as regular' queue
                                <1>
00012BD4 20E4                   <1>       and   ah, ah  ;  previous value of [u.pri]
00012BD6 75E4                   <1>       jnz   short srunseq_4
                                <1>
00012BD8 43                     <1>       inc   ebx
00012BD9 43                     <1>       inc   ebx
                                <1>       ; ebx = runq_background ; 'run on backgroud' queue
                                <1>
00012BDA 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>
00012BDC 803D[A8030300]00       <1>       cmp    byte [u.quant], 0
00012BE3 772C                   <1>       ja     short clk_1
                                <1>       ;
00012BE5 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.
00012BEC 7623                   <1>       jna    short clk_1 ; yes, do not swap out
                                <1>       ;
00012BEE 803D[5B030300]FF       <1>       cmp     byte [sysflg], 0FFh ; user or system space ?
00012BF5 7520                   <1>       jne    short clk_2      ; system space (sysflg <> 0FFh)
                                <1>       ;
00012BF7 66833D[AA030300]00     <1>       cmp    word [u.intr], 0
00012BFF 7616                   <1>       jna    short clk_2
                                <1>       ;
                                <1>       ; 23/05/2016
00012C01 803D[82950100]00       <1>       cmp    byte [multi_tasking], 0
00012C08 760D                   <1>       jna    short clk_2
                                <1>       ;
00012C0A FE05[81950100]         <1>       inc    byte [p_change] ; it is time to change running process
00012C10 C3                     <1>       retn
                                <1> clk_1:
00012C11 FE0D[A8030300]         <1>       dec    byte [u.quant]
                                <1> clk_2:
00012C17 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>       ;
                                <1>       ; Modified registers: EAX
                                <1>       ;
                                <1>
                                <1>       ;cmp   ah, 5
                                <1>       ;ja    short int34h_5 ; invalid function !
                                <1>
                                <1>       ; 12/10/2017
00012C18 80FC07                 <1>       cmp    ah, 7
00012C1B 7743                   <1>       ja     short int34h_5 ; invalid function !
                                <1>
                                <1>       ;; 15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [ss:intflg], 34h     ; IOCTL interrupt
00012C1D FB                     <1>       sti
                                <1>
                                <1>       ;sti   ; enable interrupts
00012C1E 80642408FE             <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
                                <1>
00012C23 80FC01                 <1>       cmp    ah, 1
00012C26 7205                   <1>       jb     short int34h_0
00012C28 7705                   <1>       ja     short int34h_1
                                <1>
00012C2A EE                     <1>       out    dx, al
                                <1>       ;iretd
00012C2B EB01                   <1>       jmp short int34h_iret
                                <1>
                                <1> int34h_0:
00012C2D EC                     <1>       in     al, dx
                                <1>       ;iretd
                                <1> int34h_iret:
                                <1>       ;cli   ; 07/01/2017
                                <1>       ;; 15/01/2017
                                <1>       ;;mov  byte [ss:intflg], 0 ; reset
00012C2E CF                     <1>       iretd
                                <1>
                                <1> int34h_1:
00012C2F F6C401                 <1>       test   ah, 1
00012C32 7516                   <1>       jnz    short int34h_3 ; out
```

```
3382                                    <1>
3383                                    <1>     ; in
3384 00012C34 80FC02                   <1>     cmp    ah, 2
3385 00012C37 7707                     <1>     ja     short int34h_2
3386                                    <1>
3387 00012C39 6689D8                   <1>     mov    ax, bx
3388 00012C3C 66ED                     <1>     in     ax, dx
3389                                    <1>     ;iretd
3390 00012C3E EBEE                     <1>     jmp    short int34h_iret
3391                                    <1>
3392                                    <1> int34h_2:
3393 00012C40 80FC04                   <1>     cmp    ah, 4
3394 00012C43 772C                     <1>     ja     short int34h_7      ; 12/10/2017
3395                                    <1>     ; ah = 4
3396 00012C45 89D8                     <1>     mov    eax, ebx
3397 00012C47 ED                       <1>     in     eax, dx
3398                                    <1>     ;iretd
3399 00012C48 EBE4                     <1>     jmp    short int34h_iret
3400                                    <1>
3401                                    <1> int34h_3:
3402 00012C4A 80FC03                   <1>     cmp    ah, 3
3403 00012C4D 7707                     <1>     ja     short int34h_4
3404                                    <1>
3405 00012C4F 6689D8                   <1>     mov    ax, bx
3406 00012C52 66EF                     <1>     out    dx, ax
3407                                    <1>     ;iretd
3408 00012C54 EBD8                     <1>     jmp    short int34h_iret
3409                                    <1>
3410                                    <1> int34h_4:
3411 00012C56 80FC05                   <1>     cmp    ah, 5
3412 00012C59 770B                     <1>     ja     short int34h_6      ; 12/10/2017
3413                                    <1>     ; ah = 5
3414 00012C5B 89D8                     <1>     mov    eax, ebx
3415 00012C5D EF                       <1>     out    dx, eax
3416                                    <1>     ;iretd
3417 00012C5E EBCE                     <1>     jmp    short int34h_iret
3418                                    <1>
3419                                    <1> int34h_5:
3420 00012C60 804C240801               <1>     or     byte [esp+8], 1     ; set carry bit of eflags register
3421 00012C65 CF                       <1>     iretd
3422                                    <1>
3423                                    <1>     ; 12/10/2017
3424                                    <1> int34h_6:
3425 00012C66 6689D8                   <1>     mov    ax, bx
3426 00012C69 EE                       <1>     out    dx, al
3427 00012C6A EB00                     <1>     jmp    short $+2
3428 00012C6C 86E0                     <1>     xchg   ah, al
3429 00012C6E EE                       <1>     out    dx, al
3430                                    <1>     ;xchg  al, ah
3431                                    <1>     ;iretd
3432 00012C6F EB06                     <1>     jmp    short int34h_8
3433                                    <1> int34h_7:
3434 00012C71 EC                       <1>     in     al, dx
3435 00012C72 EB00                     <1>     jmp    short $+2
3436 00012C74 88C4                     <1>     mov    ah, al
3437 00012C76 EC                       <1>     in     al, dx
3438                                    <1> int34h_8:
3439 00012C77 86C4                     <1>     xchg   al, ah
3440 00012C79 CF                       <1>     iretd
3441                                    <1>
3442                                    <1>
3443                                    <1> INT4Ah:
3444                                    <1>     ; 24/01/2016
3445                                    <1>     ; this procedure will be called by 'RTC_INT' (in 'timer.s')
3446 00012C7A C3                       <1>     retn
3447                                    <1>
3448                                    <1> ; u0.s
3449                                    <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS0.INC
3450                                    <1> ; Last Modification: 20/11/2015
3451                                    <1>
3452                                    <1> com2_int:
3453                                    <1>     ; 07/11/2015
3454                                    <1>     ; 24/10/2015
3455                                    <1>     ; 23/10/2015
3456                                    <1>     ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
3457                                    <1>     ; 28/07/2014 (Retro UNIX 8086 v1)
3458                                    <1>     ; < serial port 2 interrupt handler >
3459                                    <1>     ;
3460 00012C7B 890424                   <1>     mov    [esp], eax ; overwrite call return address
3461                                    <1>     ;push  eax
3462 00012C7E 66B80900                 <1>     mov    ax, 9
3463 00012C82 EB07                     <1>     jmp    short comm_int
3464                                    <1> com1_int:
3465                                    <1>     ; 07/11/2015
3466                                    <1>     ; 24/10/2015
3467 00012C84 890424                   <1>     mov    [esp], eax ; overwrite call return address
3468                                    <1>     ; 23/10/2015
3469                                    <1>     ;push  eax
3470 00012C87 66B80800                 <1>     mov    ax, 8
3471                                    <1> comm_int:
3472                                    <1>     ; 20/11/2015
3473                                    <1>     ; 18/11/2015
3474                                    <1>     ; 17/11/2015
3475                                    <1>     ; 16/11/2015
3476                                    <1>     ; 09/11/2015
3477                                    <1>     ; 08/11/2015
3478                                    <1>     ; 07/11/2015
3479                                    <1>     ; 06/11/2015 (serial4.asm, 'serial')
3480                                    <1>     ; 01/11/2015
3481                                    <1>     ; 26/10/2015
3482                                    <1>     ; 23/10/2015
3483 00012C8B 53                       <1>     push   ebx
3484 00012C8C 56                       <1>     push   esi
3485 00012C8D 57                       <1>     push   edi
3486 00012C8E 1E                       <1>     push   ds
```

```
3487 00012C8F 06                <1>       push   es
3488                            <1>       ; 18/11/2015
3489 00012C90 0F20DB            <1>       mov    ebx, cr3
3490 00012C93 53                <1>       push   ebx ; ****
3491                            <1>       ;
3492 00012C94 51                <1>       push   ecx ; ***
3493 00012C95 52                <1>       push   edx ; **
3494                            <1>       ;
3495 00012C96 BB10000000        <1>       mov    ebx, KDATA
3496 00012C9B 8EDB              <1>       mov    ds, bx
3497 00012C9D 8EC3              <1>       mov    es, bx
3498                            <1>       ;
3499 00012C9F 8B0D[F0880100]    <1>       mov    ecx, [k_page_dir]
3500 00012CA5 0F22D9            <1>       mov    cr3, ecx
3501                            <1>       ; 20/11/2015
3502                            <1>       ; Interrupt identification register
3503 00012CA8 66BAFA02          <1>       mov    dx, 2FAh ; COM2
3504                            <1>       ;
3505 00012CAC 3C08              <1>       cmp    al, 8
3506 00012CAE 7702              <1>       ja     short com_i0
3507                            <1>       ;
3508                            <1>       ; 20/11/2015
3509                            <1>       ; 17/11/2015
3510                            <1>       ; 16/11/2015
3511                            <1>       ; 15/11/2015
3512                            <1>       ; 24/10/2015
3513                            <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
3514                            <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
3515                            <1>       ; < serial port 1 interrupt handler >
3516                            <1>       ;
3517 00012CB0 FEC6              <1>       inc    dh ; 3FAh ; COM1 Interrupt id. register
3518                            <1> com_i0:
3519                            <1>       ;push  eax ; *
3520                            <1>       ; 07/11/2015
3521 00012CB2 A2[5A890100]      <1>       mov    byte [ccomport], al
3522                            <1>       ; 09/11/2015
3523 00012CB7 0FB7D8            <1>       movzx  ebx, ax ; 8 or 9
3524                            <1>       ; 17/11/2015
3525                            <1>       ; reset request for response status
3526 00012CBA 88A3[50890100]    <1>       mov    [ebx+req_resp-8], ah ; 0
3527                            <1>       ;
3528                            <1>       ; 20/11/2015
3529 00012CC0 EC                <1>       in     al, dx       ; read interrupt id. register
3530 00012CC1 EB00              <1>       JMP    $+2          ; I/O DELAY
3531 00012CC3 2404              <1>       and    al, 4        ; received data available?
3532 00012CC5 7470              <1>       jz     short com_eoi; (transmit. holding reg. empty)
3533                            <1>       ;
3534                            <1>       ; 20/11/2015
3535 00012CC7 80EA02            <1>       sub    dl, 3FAh-3F8h; data register (3F8h, 2F8h)
3536 00012CCA EC                <1>       in     al, dx       ; read character
3537                            <1>       ;JMP   $+2          ; I/O DELAY
3538                            <1>       ; 08/11/2015
3539                            <1>       ; 07/11/2015
3540 00012CCB 89DE              <1>       mov    esi, ebx
3541 00012CCD 89DF              <1>       mov    edi, ebx
3542 00012CCF 81C6[54890100]    <1>       add    esi, rchar - 8 ; points to last received char
3543 00012CD5 81C7[56890100]    <1>       add    edi, schar - 8 ; points to last sent char
3544 00012CDB 8806              <1>       mov    [esi], al ; received char (current char)
3545                            <1>       ; query
3546 00012CDD 20C0              <1>       and    al, al
3547 00012CDF 7527              <1>       jnz    short com_i2
3548                            <1>       ; response
3549                            <1>       ; 17/11/2015
3550                            <1>       ; set request for response status
3551 00012CE1 FE83[50890100]    <1>         inc     byte [ebx+req_resp-8] ; 1
3552                            <1>       ;
3553 00012CE7 6683C205          <1>       add    dx, 3FDh-3F8h; (3FDh, 2FDh)
3554 00012CEB EC                <1>       in     al, dx       ; read line status register
3555 00012CEC EB00              <1>       JMP    $+2          ; I/O DELAY
3556 00012CEE 2420              <1>       and    al, 20h             ; transmitter holding reg. empty?
3557 00012CF0 7445              <1>       jz     short com_eoi       ; no
3558 00012CF2 B0FF              <1>       mov    al, 0FFh     ; response
3559 00012CF4 6683EA05          <1>       sub    dx, 3FDh-3F8h       ; data port (3F8h, 2F8h)
3560 00012CF8 EE                <1>       out    dx, al       ; send on serial port
3561                            <1>       ; 17/11/2015
3562 00012CF9 803F00            <1>       cmp    byte [edi], 0   ; query ? (schar)
3563 00012CFC 7502              <1>       jne    short com_i1    ; no
3564 00012CFE 8807              <1>       mov    [edi], al    ; 0FFh (responded)
3565                            <1> com_i1:
3566                            <1>       ; 17/11/2015
3567                            <1>       ; reset request for response status (again)
3568 00012D00 FE8B[50890100]    <1>         dec     byte [ebx+req_resp-8] ; 0
3569 00012D06 EB2F              <1>       jmp    short com_eoi
3570                            <1> com_i2:
3571                            <1>       ; 08/11/2015
3572 00012D08 3CFF              <1>       cmp    al, 0FFh     ; (response ?)
3573 00012D0A 7417              <1>       je     short com_i3 ; (check for response signal)
3574                            <1>       ; 07/11/2015
3575 00012D0C 3C04              <1>       cmp    al, 04h      ; EOT
3576 00012D0E 751C              <1>       jne    short com_i4
3577                            <1>       ; EOT = 04h (End of Transmit) - 'CTRL + D'
3578                            <1>       ;(an EOT char is supposed as a ctrl+brk from the terminal)
3579                            <1>       ; 08/11/2015
3580                            <1>             ; ptty -> tty 0 to 7 (pseudo screens)
3581 00012D10 861D[1E890100]    <1>       xchg   bl, [ptty]  ; tty number (8 or 9)
3582 00012D16 E89547FFFF        <1>       call   ctrlbrk
3583 00012D1B 861D[1E890100]    <1>       xchg   [ptty], bl ; (restore ptty value and BL value)
3584                            <1>       ;mov   al, 04h ; EOT
3585                            <1>       ; 08/11/2015
3586 00012D21 EB09              <1>       jmp    short com_i4
3587                            <1> com_i3:
3588                            <1>       ; 08/11/2015
3589                            <1>       ; If 0FFh has been received just after a query
3590                            <1>       ; (schar, ZERO), it is a response signal.
3591                            <1>       ; 17/11/2015
```

```
3592 00012D23 803F00              <1>       cmp   byte [edi], 0 ; query ? (schar)
3593 00012D26 7704                <1>       ja    short com_i4 ; no
3594                              <1>       ; reset query status (schar)
3595 00012D28 8807                <1>       mov   [edi], al ; 0FFh
3596 00012D2A FEC0                <1>       inc   al ; 0
3597                              <1> com_i4:
3598                              <1>       ; 27/07/2014
3599                              <1>       ; 09/07/2014
3600 00012D2C D0E3                <1>       shl   bl, 1
3601 00012D2E 81C3[20890100]      <1>       add   ebx, ttychr
3602                              <1>       ; 23/07/2014 (always overwrite)
3603                              <1>       ;;cmp  word [ebx], 0
3604                              <1>       ;;ja   short com_eoi
3605                              <1>       ;
3606 00012D34 668903              <1>       mov   [ebx], ax   ; Save ascii code
3607                              <1>                         ; scan code = 0
3608                              <1> com_eoi:
3609                              <1>       ;mov   al, 20h
3610                              <1>       ;out   20h, al         ; end of interrupt
3611                              <1>       ;
3612                              <1>       ; 07/11/2015
3613                              <1>       ;pop   eax ; *
3614 00012D37 A0[5A890100]        <1>       mov   al, byte [ccomport] ; current COM port
3615                              <1>       ; al = tty number (8 or 9)
3616 00012D3C E85E010000          <1>       call  wakeup
3617                              <1> com_iret:
3618                              <1>       ; 23/10/2015
3619 00012D41 5A                  <1>       pop   edx ; **
3620 00012D42 59                  <1>       pop   ecx ; ***
3621                              <1>       ; 18/11/2015
3622                              <1>       ;pop   eax ; ****
3623                              <1>       ;mov   cr3, eax
3624                              <1>       ;jmp   iiret
3625 00012D43 E961E0FEFF          <1>       jmp   iiretp
3626                              <1>
3627                              <1> ;iiretp: ; 01/09/2015
3628                              <1> ;     ; 28/08/2015
3629                              <1> ;     pop   eax ; (*) page directory
3630                              <1> ;     mov   cr3, eax
3631                              <1> ;iiret:
3632                              <1> ;     ; 22/08/2014
3633                              <1> ;     mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
3634                              <1> ;     out   20h, al      ; 8259 PORT
3635                              <1> ;     ;
3636                              <1> ;     pop   es
3637                              <1> ;     pop   ds
3638                              <1> ;     pop   edi
3639                              <1> ;     pop   esi
3640                              <1> ;     pop   ebx ; 29/08/2014
3641                              <1> ;     pop   eax
3642                              <1> ;     iretd
3643                              <1>
3644                              <1> sp_init:
3645                              <1>       ; 07/11/2015
3646                              <1>       ; 29/10/2015
3647                              <1>       ; 26/10/2015
3648                              <1>       ; 23/10/2015
3649                              <1>       ; 29/06/2015
3650                              <1>       ; 14/03/2015 (Retro UNIX 386 v1 - 115200 baud)
3651                              <1>       ; 28/07/2014 (Retro UNIX 8086 v1 - 9600 baud)
3652                              <1>       ; Initialization of Serial Port Communication Parameters
3653                              <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
3654                              <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
3655                              <1>       ;
3656                              <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
3657                              <1>       ;
3658                              <1>       ; INPUT:  (29/06/2015)
3659                              <1>       ;     AL = 0 for COM1
3660                              <1>       ;          1 for COM2
3661                              <1>       ;     AH = Communication parameters
3662                              <1>       ;
3663                              <1>       ;  (*) Communication parameters (except BAUD RATE):
3664                              <1>       ;     Bit    4     3     2     1     0
3665                              <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
3666                              <1>       ;  this one -->    00 = none    0 = 1 bit  11 = 8 bits
3667                              <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
3668                              <1>       ;            11 = even
3669                              <1>       ;  Baud rate setting bits: (29/06/2015)
3670                              <1>       ;            Retro UNIX 386 v1 feature only !
3671                              <1>       ;     Bit    7    6    5  | Baud rate
3672                              <1>       ;            ------------------------
3673                              <1>       ;     value  0    0    0  | Default (Divisor = 1)
3674                              <1>       ;            0    0    1  | 9600 (12)
3675                              <1>       ;            0    1    0  | 19200 (6)
3676                              <1>       ;            0    1   1  | 38400 (3)
3677                              <1>       ;            1    0   0  | 14400 (8)
3678                              <1>       ;            1    0   1  | 28800 (4)
3679                              <1>       ;            1    1    0  | 57600 (2)
3680                              <1>       ;            1    1    1  | 115200 (1)
3681                              <1>
3682                              <1>       ; References:
3683                              <1>       ; (1) IBM PC-XT Model 286 BIOS Source Code
3684                              <1>       ;     RS232.ASM --- 10/06/1985 COMMUNICATIONS BIOS (RS232)
3685                              <1>       ; (2) Award BIOS 1999 - ATORGS.ASM
3686                              <1>       ; (3) http://wiki.osdev.org/Serial_Ports
3687                              <1>       ;
3688                              <1>       ; Set communication parameters for COM1 (= 03h)
3689                              <1>       ;
3690 00012D48 BB[56890100]        <1>       mov   ebx, com1p          ; COM1 parameters
3691 00012D4D 66BAF803            <1>       mov   dx, 3F8h            ; COM1
3692                              <1>        ; 29/10/2015
3693 00012D51 66B90103            <1>       mov   cx, 301h  ; divisor = 1 (115200 baud)
3694 00012D55 E86F000000          <1>       call  sp_i3  ; call A4
3695 00012D5A A880                <1>       test  al, 80h
3696 00012D5C 7410                <1>       jz    short sp_i0 ; OK..
```

```
3697                            <1>       ; Error !
3698                            <1>       ;mov   dx, 3F8h
3699 00012D5E 80EA05            <1>       sub    dl, 5 ; 3FDh -> 3F8h
3700 00012D61 66B90E03          <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
3701 00012D65 E85F000000        <1>       call   sp_i3  ; call A4
3702 00012D6A A880              <1>       test   al, 80h
3703 00012D6C 7508              <1>       jnz    short sp_i1
3704                            <1> sp_i0:
3705                            <1>       ; (Note: Serial port interrupts will be disabled here...)
3706                            <1>       ; (INT 14h initialization code disables interrupts.)
3707                            <1>       ;
3708 00012D6E C603E3            <1>       mov    byte [ebx], 0E3h ; 11100011b
3709 00012D71 E8DC000000        <1>       call   sp_i5 ; 29/06/2015
3710                            <1> sp_i1:
3711 00012D76 43                <1>       inc    ebx
3712 00012D77 66BAF802          <1>       mov    dx, 2F8h            ; COM2
3713                            <1>       ; 29/10/2015
3714 00012D7B 66B90103          <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
3715 00012D7F E845000000        <1>       call   sp_i3  ; call A4
3716 00012D84 A880              <1>       test   al, 80h
3717 00012D86 7410              <1>       jz     short sp_i2 ; OK..
3718                            <1>       ; Error !
3719                            <1>       ;mov   dx, 2F8h
3720 00012D88 80EA05            <1>       sub    dl, 5 ; 2FDh -> 2F8h
3721 00012D8B 66B90E03          <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
3722 00012D8F E835000000        <1>       call   sp_i3  ; call A4
3723 00012D94 A880              <1>       test   al, 80h
3724 00012D96 7530              <1>       jnz    short sp_i7
3725                            <1> sp_i2:
3726 00012D98 C603E3            <1>       mov    byte [ebx], 0E3h ; 11100011b
3727                            <1> sp_i6:
3728                            <1>       ;; COM2 - enabling IRQ 3
3729                            <1>       ; 07/11/2015
3730                            <1>       ; 26/10/2015
3731 00012D9B 9C                <1>       pushf
3732 00012D9C FA                <1>       cli
3733                            <1>       ;
3734 00012D9D 66BAFC02          <1>       mov    dx, 2FCh            ; modem control register
3735 00012DA1 EC                <1>       in     al, dx                   ; read register
3736 00012DA2 EB00              <1>       JMP    $+2                 ; I/O DELAY
3737 00012DA4 0C08              <1>       or     al, 8               ; enable bit 3 (OUT2)
3738 00012DA6 EE                <1>       out    dx, al              ; write back to register
3739 00012DA7 EB00              <1>       JMP    $+2                 ; I/O DELAY
3740 00012DA9 66BAF902          <1>       mov    dx, 2F9h            ; interrupt enable register
3741 00012DAD EC                <1>       in     al, dx              ; read register
3742 00012DAE EB00              <1>       JMP    $+2                 ; I/O DELAY
3743                            <1>       ;or    al, 1               ; receiver data interrupt enable and
3744 00012DB0 0C03              <1>       or     al, 3               ; transmitter empty interrupt enable
3745 00012DB2 EE                <1>       out    dx, al                   ; write back to register
3746 00012DB3 EB00              <1>       JMP    $+2                 ; I/O DELAY
3747 00012DB5 E421              <1>       in     al, 21h             ; read interrupt mask register
3748 00012DB7 EB00              <1>       JMP    $+2                 ; I/O DELAY
3749 00012DB9 24F7              <1>       and    al, 0F7h            ; enable IRQ 3 (COM2)
3750 00012DBB E621              <1>       out    21h, al             ; write back to register
3751                            <1>       ;
3752                            <1>       ; 23/10/2015
3753 00012DBD B8[7B2C0100]      <1>       mov    eax, com2_int
3754 00012DC2 A3[9A2E0100]      <1>       mov    [com2_irq3], eax
3755                            <1>       ; 26/10/2015
3756 00012DC7 9D                <1>       popf
3757                            <1> sp_i7:
3758 00012DC8 C3                <1>       retn
3759                            <1>
3760                            <1> sp_i3:
3761                            <1> ;A4:        ;----- INITIALIZE THE COMMUNICATIONS PORT
3762                            <1>       ; 28/10/2015
3763 00012DC9 FEC2              <1>       inc    dl     ; 3F9h (2F9h); 3F9h, COM1 Interrupt enable register
3764 00012DCB B000              <1>       mov    al, 0
3765 00012DCD EE                <1>       out    dx, al              ; disable serial port interrupt
3766 00012DCE EB00              <1>       JMP    $+2                 ; I/O DELAY
3767 00012DD0 80C202            <1>       add    dl, 2  ; 3FBh (2FBh); COM1 Line control register (3FBh)
3768 00012DD3 B080              <1>       mov    al, 80h
3769 00012DD5 EE                <1>       out    dx, al              ; SET DLAB=1 ; divisor latch access bit
3770                            <1>       ;----- SET BAUD RATE DIVISOR
3771                            <1>       ; 26/10/2015
3772 00012DD6 80EA03            <1>       sub    dl, 3   ; 3F8h (2F8h)      ; register for least significant byte
3773                            <1>                                   ; of the divisor value
3774 00012DD9 88C8              <1>       mov    al, cl ; 1
3775 00012DDB EE                <1>       out    dx, al              ; 1 = 115200 baud (Retro UNIX 386 v1)
3776                            <1>                                   ; 2 = 57600 baud
3777                            <1>                                   ; 3 = 38400 baud
3778                            <1>                                   ; 6 = 19200 baud
3779                            <1>                                   ; 12 = 9600 baud (Retro UNIX 8086 v1)
3780 00012DDC EB00              <1>       JMP    $+2                 ; I/O DELAY
3781 00012DDE 28C0              <1>       sub    al, al
3782 00012DE0 FEC2              <1>       inc    dl      ; 3F9h (2F9h)      ; register for most significant byte
3783                            <1>                                   ; of the divisor value
3784 00012DE2 EE                <1>       out    dx, al ; 0
3785 00012DE3 EB00              <1>       JMP    $+2                 ; I/O DELAY
3786                            <1>       ;
3787 00012DE5 88E8              <1>       mov    al, ch ; 3          ; 8 data bits, 1 stop bit, no parity
3788                            <1>       ;and   al, 1Fh ; Bits 0,1,2,3,4
3789 00012DE7 80C202            <1>       add    dl, 2  ; 3FBh (2FBh); Line control register
3790 00012DEA EE                <1>       out    dx, al
3791 00012DEB EB00              <1>       JMP    $+2                 ; I/O DELAY
3792                            <1>       ; 29/10/2015
3793 00012DED FECA              <1>       dec    dl     ; 3FAh (2FAh); FIFO Control register (16550/16750)
3794 00012DEF 30C0              <1>       xor    al, al              ; 0
3795 00012DF1 EE                <1>       out    dx, al              ; Disable FIFOs (reset to 8250 mode)
3796 00012DF2 EB00              <1>       JMP    $+2
3797                            <1> sp_i4:
3798                            <1> ;A18: ;----- COMM PORT STATUS ROUTINE
3799                            <1>       ; 29/06/2015 (line status after modem status)
3800 00012DF4 80C204            <1>       add    dl, 4  ; 3FEh (2FEh); Modem status register
3801                            <1> sp_i4s:
```

```
3802 00012DF7 EC                <1>       in     al, dx              ; GET MODEM CONTROL STATUS
3803 00012DF8 EB00              <1>       JMP    $+2                 ; I/O DELAY
3804 00012DFA 88C4              <1>       mov    ah, al              ; PUT IN (AH) FOR RETURN
3805 00012DFC FECA              <1>       dec    dl     ; 3FDh (2FDh); POINT TO LINE STATUS REGISTER
3806                            <1>                                  ; dx = 3FDh for COM1, 2FDh for COM2
3807 00012DFE EC                <1>       in     al, dx              ; GET LINE CONTROL STATUS
3808                            <1>       ; AL = Line status, AH = Modem status
3809 00012DFF C3                <1>       retn
3810                            <1>
3811                            <1> sp_status:
3812                            <1>       ; 29/06/2015
3813                            <1>       ; 27/06/2015 (Retro UNIX 386 v1)
3814                            <1>       ; Get serial port status
3815 00012E00 66BAFE03          <1>       mov    dx, 3FEh             ; Modem status register (COM1)
3816 00012E04 28C6              <1>       sub    dh, al               ; dh = 2 for COM2 (al = 1)
3817                            <1>                                   ; dx = 2FEh for COM2
3818 00012E06 EBEF              <1>       jmp    short sp_i4s
3819                            <1>
3820                            <1> sp_setp: ; Set serial port communication parameters
3821                            <1>       ; 07/11/2015
3822                            <1>       ; 29/10/2015
3823                            <1>       ; 29/06/2015
3824                            <1>       ; Retro UNIX 386 v1 feature only !
3825                            <1>       ;
3826                            <1>       ; INPUT:
3827                            <1>       ;      AL = 0 for COM1
3828                            <1>       ;           1 for COM2
3829                            <1>       ;      AH = Communication parameters (*)
3830                            <1>       ; OUTPUT:
3831                            <1>       ;      CL = Line status
3832                            <1>       ;      CH = Modem status
3833                            <1>       ;   If cf = 1 -> Error code in [u.error]
3834                            <1>       ;              'invalid parameter !'
3835                            <1>       ;                   or
3836                            <1>       ;             'device not ready !' error
3837                            <1>       ;
3838                            <1>       ;  (*) Communication parameters (except BAUD RATE):
3839                            <1>       ;     Bit    4      3      2      1      0
3840                            <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
3841                            <1>       ;  this one -->     00 = none    0 = 1 bit  11 = 8 bits
3842                            <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
3843                            <1>       ;            11 = even
3844                            <1>       ;  Baud rate setting bits: (29/06/2015)
3845                            <1>       ;            Retro UNIX 386 v1 feature only !
3846                            <1>       ;      Bit    7    6    5  | Baud rate
3847                            <1>       ;            ------------------------
3848                            <1>       ;      value  0    0    0  | Default (Divisor = 1)
3849                            <1>       ;            0    0    1  | 9600 (12)
3850                            <1>       ;            0    1    0  | 19200 (6)
3851                            <1>       ;            0    1   1   | 38400 (3)
3852                            <1>       ;            1    0   0   | 14400 (8)
3853                            <1>       ;            1    0   1   | 28800 (4)
3854                            <1>       ;            1    1    0  | 57600 (2)
3855                            <1>       ;            1    1    1  | 115200 (1)
3856                            <1>       ;
3857                            <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
3858                            <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
3859                            <1>       ;
3860                            <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
3861                            <1>       ;
3862 00012E08 66BAF803          <1>       mov    dx, 3F8h
3863 00012E0C BB[56890100]      <1>       mov    ebx, com1p ; COM1 control byte offset
3864 00012E11 3C01              <1>       cmp    al, 1
3865 00012E13 776B              <1>       ja     short sp_invp_err
3866 00012E15 7203              <1>       jb     short sp_setp1 ;  COM1 (AL = 0)
3867 00012E17 FECE              <1>       dec    dh ; 2F8h
3868 00012E19 43                <1>       inc    ebx ; COM2 control byte offset
3869                            <1> sp_setp1:
3870                            <1>       ; 29/10/2015
3871 00012E1A 8823              <1>       mov    [ebx], ah
3872 00012E1C 0FB6CC            <1>       movzx  ecx, ah
3873 00012E1F C0E905            <1>       shr    cl, 5 ; -> baud rate index
3874 00012E22 80E41F            <1>       and    ah, 1Fh ; communication parameters except baud rate
3875 00012E25 8A81[8F2E0100]    <1>       mov    al, [ecx+b_div_tbl]
3876 00012E2B 6689C1            <1>       mov    cx, ax
3877 00012E2E E896FFFFFF        <1>       call   sp_i3
3878 00012E33 6689C1            <1>       mov    cx, ax ; CL = Line status, CH = Modem status
3879 00012E36 A880              <1>       test   al, 80h
3880 00012E38 740F              <1>       jz     short sp_setp2
3881 00012E3A C603E3            <1>         mov     byte [ebx], 0E3h ; Reset to initial value (11100011b)
3882                            <1> stp_dnr_err:
3883 00012E3D C705[C8030300]0F00- <1>     mov    dword [u.error], ERR_DEV_NOT_RDY ; 'device not ready !'
3883 00012E45 0000              <1>
3884                            <1>       ; CL = Line status, CH = Modem status
3885 00012E47 F9                <1>       stc
3886 00012E48 C3                <1>       retn
3887                            <1> sp_setp2:
3888 00012E49 80FE02            <1>       cmp    dh, 2 ; COM2 (2F?h)
3889 00012E4C 0F8649FFFFFF      <1>       jna   sp_i6
3890                            <1>                   ; COM1 (3F?h)
3891                            <1> sp_i5:
3892                            <1>       ; 07/11/2015
3893                            <1>       ; 26/10/2015
3894                            <1>       ; 29/06/2015
3895                            <1>       ;
3896                            <1>       ;; COM1 - enabling IRQ 4
3897 00012E52 9C                <1>       pushf
3898 00012E53 FA                <1>       cli
3899 00012E54 66BAFC03          <1>       mov    dx, 3FCh            ; modem control register
3900 00012E58 EC                <1>       in     al, dx                    ; read register
3901 00012E59 EB00              <1>       JMP    $+2                 ; I/O DELAY
3902 00012E5B 0C08              <1>       or     al, 8               ; enable bit 3 (OUT2)
3903 00012E5D EE                <1>       out    dx, al              ; write back to register
3904 00012E5E EB00              <1>       JMP    $+2                 ; I/O DELAY
3905 00012E60 66BAF903          <1>       mov    dx, 3F9h            ; interrupt enable register
```

```
3906 00012E64 EC                  <1>       in     al, dx              ; read register
3907 00012E65 EB00                <1>       JMP    $+2                 ; I/O DELAY
3908                              <1>       ;or    al, 1               ; receiver data interrupt enable and
3909 00012E67 0C03                <1>       or     al, 3               ; transmitter empty interrupt enable
3910 00012E69 EE                  <1>       out    dx, al                    ; write back to register
3911 00012E6A EB00                <1>       JMP    $+2                 ; I/O DELAY
3912 00012E6C E421                <1>       in     al, 21h             ; read interrupt mask register
3913 00012E6E EB00                <1>       JMP    $+2                 ; I/O DELAY
3914 00012E70 24EF                <1>       and    al, 0EFh            ; enable IRQ 4 (COM1)
3915 00012E72 E621                <1>       out    21h, al             ; write back to register
3916                              <1>       ;
3917                              <1>       ; 23/10/2015
3918 00012E74 B8[842C0100]        <1>       mov    eax, com1_int
3919 00012E79 A3[962E0100]        <1>       mov    [com1_irq4], eax
3920                              <1>       ; 26/10/2015
3921 00012E7E 9D                  <1>       popf
3922 00012E7F C3                  <1>       retn
3923                              <1>
3924                              <1> sp_invp_err:
3925 00012E80 C705[C8030300]1700- <1>       mov    dword [u.error], ERR_INV_PARAMETER ; 'invalid parameter !'
3925 00012E88 0000                <1>
3926 00012E8A 31C9                <1>       xor    ecx, ecx
3927 00012E8C 49                  <1>       dec    ecx ; 0FFFFh
3928 00012E8D F9                  <1>       stc
3929 00012E8E C3                  <1>       retn
3930                              <1>
3931                              <1> ; 29/10/2015
3932                              <1> b_div_tbl: ; Baud rate divisor table (115200/divisor)
3933 00012E8F 010C0603080401      <1>       db 1, 12, 6, 3, 8, 4, 1
3934                              <1>
3935                              <1>
3936                              <1> ; 23/10/2015
3937                              <1> com1_irq4:
3938 00012E96 [9E2E0100]          <1>       dd dummy_retn
3939                              <1> com2_irq3:
3940 00012E9A [9E2E0100]          <1>       dd dummy_retn
3941                              <1>
3942                              <1> dummy_retn:
3943 00012E9E C3                  <1>       retn
3944                              <1>
3945                              <1> wakeup:
3946                              <1>       ; 24/01/2016
3947 00012E9F C3                  <1>       retn
3948                              <1>
3949                              <1> set_working_path_x:
3950                              <1>             ; 17/10/2016 (TRDOS 386 - FFF & FNF)
3951 00012EA0 66B80100            <1>             mov    ax, 1
3952                              <1>             ; File name is needed/forced (AL=1)
3953                              <1>             ; Change directory as temporary (AH=0)
3954                              <1>
3955                              <1> set_working_path_xx: ; 30/12/2017 (syschdir)
3956                              <1>             ; This is needed for preventing wrong Find Next File
3957                              <1>             ; system call after sysopen, syscreate, sysmkdir etc.
3958                              <1>             ; Find Next File must immediate follow Find First File)
3959                              <1>
3960 00012EA4 8825[A4950100]      <1>             mov    [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
3961                              <1>
3962                              <1> set_working_path:
3963                              <1>             ; 16/10/2016
3964                              <1>             ; 12/10/2016
3965                              <1>             ; 10/10/2016
3966                              <1>             ; 05/10/2016 - TRDOS 386 (TRDOS v2.0)
3967                              <1>             ;
3968                              <1>             ; TRDOS v1.0 (DIR.ASM, "proc_set_working_path")
3969                              <1>                ; 27/01/2011 - 08/02/2011
3970                              <1>             ; Set/Changes current drive, directory and file
3971                              <1>             ; depending on command tail
3972                              <1>             ; (procedure is derivated from CMD_INTR.ASM
3973                              <1>             ; file or dir locating code of internal commands)
3974                              <1>             ; (This procedure is prepared for INT 21H file/dir
3975                              <1>             ; functions and also to get compact code for
3976                              <1>             ; internal mainprog -command interpreter- commands)
3977                              <1>             ;
3978                              <1>             ; INPUT: DS:SI -> Command tail (ASCIIZ string)
3979                              <1>             ; AL = 0 -> any, AL > 0 -> file name is forced
3980                              <1>             ; AH = CD -> Change directory permanently
3981                              <1>             ; AH <> CD -> Change directory as temporary
3982                              <1>             ;
3983                              <1>             ; OUTPUT: ES=DS, FindFile structure has been set
3984                              <1>             ;        RUN_CDRV points previous current drive
3985                              <1>             ;        DS:SI = FindFile structure address
3986                              <1>             ;        (DS=CS)
3987                              <1>             ;        AX, BX, CX, DX, DI will be changed
3988                              <1>             ;   cf = 1 -> Error code in AX (AL)
3989                              <1>             ;        stc & AX = 0 -> Bad command or path name
3990                              <1>             ; ------------------------------------------------
3991                              <1>             ;
3992                              <1>             ; TRDOS 386 (05/10/2016)
3993                              <1>             ; INPUT:
3994                              <1>             ;     ESI = File/Directory Path (ASCIIZ string)
3995                              <1>             ;             address in user's memory space
3996                              <1>             ;       AL = 0 -> any
3997                              <1>             ;       AL > 0 -> file name is forced
3998                              <1>             ;       AH = CD -> change directory as permanent
3999                              <1>             ;       AH <> CD -> change directory as temporary
4000                              <1>             ;
4001                              <1>             ; OUTPUT:
4002                              <1>             ;     FindFile structure has been set
4003                              <1>             ;       RUN_CDRV points previous current drive
4004                              <1>             ;       ESI = FindFile_Name address ; 12/10/2016
4005                              <1>             ;
4006                              <1>             ;       cf = 1 -> Error code in EAX (AL)
4007                              <1>             ;       stc & EAX = 0 -> Bad command or path name
4008                              <1>             ;
4009                              <1>             ; Modified registers: EAX, EBX, ECX, EDX, ESI, EDI
```

```
4010                                    <1>
4011 00012EAA 66A3[A8950100]            <1>             mov    [SWP_Mode], ax
4012 00012EB0 A0[B6890100]              <1>             mov    al, [Current_Drv]
4013 00012EB5 30E4                      <1>             xor    ah, ah
4014 00012EB7 66A3[AA950100]            <1>             mov    [SWP_DRV], ax
4015                                    <1>
4016                                    <1>             ; TRDOS 386 ring 3 (user's page directory)
4017                                    <1>             ; to ring 0 (kernel's page directory)
4018                                    <1>             ; transfer modifications (05/10/2016).
4019                                    <1>
4020 00012EBD 55                        <1>             push   ebp
4021 00012EBE 89E5                      <1>             mov    ebp, esp
4022                                    <1>
4023 00012EC0 B980000000                <1>             mov    ecx, 128 ; maximum path length = 128 bytes
4024 00012EC5 29CC                      <1>             sub    esp, ecx ; reserve 128 bytes (buffer) on stack
4025 00012EC7 89E7                      <1>             mov    edi, esp ; destination address (kernel space)
4026                                    <1>             ; esi = source address (virtual, in user's memory space)
4027 00012EC9 E8A0EBFFFF                <1>             call   transfer_from_user_buffer
4028 00012ECE 720A                      <1>             jc     short loc_swp_xor_retn
4029                                    <1>
4030 00012ED0 89E6                      <1>             mov    esi, esp ; temporary buffer (the path) on stack
4031                                    <1> loc_swp_fchar:
4032 00012ED2 8A06                      <1>             mov    al, [esi]
4033 00012ED4 3C20                      <1>             cmp    al, 20h
4034 00012ED6 7711                      <1>             ja     short loc_swp_parse_path_name
4035 00012ED8 740C                      <1>             je     short loc_swp_fchar_next
4036                                    <1>
4037                                    <1> loc_swp_xor_retn:
4038 00012EDA 31C0                      <1>             xor    eax, eax
4039 00012EDC F9                        <1>             stc
4040                                    <1> loc_swp_retn:
4041 00012EDD 89EC                      <1>             mov    esp, ebp
4042 00012EDF 5D                        <1>             pop    ebp
4043                                    <1>
4044                                    <1>             ;mov   esi, FindFile_Drv
4045 00012EE0 BE[98920100]              <1>             mov    esi, FindFile_Name ; 12/10/2016
4046 00012EE5 C3                        <1>             retn
4047                                    <1>
4048                                    <1> loc_swp_fchar_next:
4049 00012EE6 46                        <1>             inc    esi
4050 00012EE7 EBE9                      <1>             jmp    short loc_swp_fchar
4051                                    <1>
4052                                    <1> loc_swp_parse_path_name:
4053 00012EE9 BF[56920100]              <1>             mov    edi, FindFile_Drv
4054 00012EEE E82F86FFFF                <1>             call   parse_path_name
4055 00012EF3 72E8                      <1>             jc     short loc_swp_retn
4056                                    <1>
4057                                    <1> loc_swp_checkfile_name:
4058 00012EF5 803D[A8950100]00          <1>             cmp    byte [SWP_Mode], 0
4059 00012EFC 761E                      <1>             jna    short loc_swp_drv
4060                                    <1>
4061                                    <1>             ; 10/10/2016 (valid file name checking)
4062 00012EFE BE[98920100]              <1>             mov    esi, FindFile_Name
4063 00012F03 803E20                    <1>             cmp    byte [esi], 20h
4064 00012F06 76D2                      <1>             jna    short loc_swp_xor_retn
4065                                    <1>
4066                                    <1>             ; 16/10/2016
4067 00012F08 C605[A7950100]00          <1>             mov    byte [SWP_inv_fname], 0 ; reset
4068                                    <1>             ; esi = file name address (ASCIIZ)
4069 00012F0F E8FB67FFFF                <1>             call   check_filename
4070 00012F14 7306                      <1>             jnc    short loc_swp_drv
4071                                    <1>
4072 00012F16 FE05[A7950100]            <1>             inc    byte [SWP_inv_fname] ; set
4073                                    <1> loc_swp_drv:
4074 00012F1C 8A35[B6890100]            <1>             mov    dh, [Current_Drv]
4075                                    <1>               ;mov      [RUN_CDRV], dh
4076                                    <1>
4077 00012F22 8A15[56920100]            <1>             mov    dl, [FindFile_Drv]
4078                                    <1>               ;cmp      dl, dh
4079 00012F28 3A15[B6890100]            <1>             cmp    dl, [Current_Drv]
4080 00012F2E 740D                      <1>             je     short loc_swp_change_directory
4081                                    <1>
4082 00012F30 FE05[AB950100]            <1>             inc    byte [SWP_DRV_chg]
4083 00012F36 E87350FFFF                <1>             call   change_current_drive
4084 00012F3B 72A0                      <1>             jc     short loc_swp_retn ; eax = error code
4085                                    <1>             ; eax = 0
4086                                    <1>
4087                                    <1> loc_swp_change_directory:
4088 00012F3D 803D[57920100]21          <1>             cmp    byte [FindFile_Directory], 21h
4089 00012F44 F5                        <1>             cmc
4090 00012F45 7396                      <1>             jnc    short loc_swp_retn
4091                                    <1>
4092 00012F47 FE05[AB950100]            <1>             inc    byte [SWP_DRV_chg]
4093 00012F4D FE05[833F0100]            <1>             inc    byte [Restore_CDIR]
4094 00012F53 BE[57920100]              <1>             mov    esi, FindFile_Directory
4095 00012F58 8A25[A9950100]            <1>             mov    ah, [SWP_Mode+1]
4096 00012F5E E8A97FFFFF                <1>             call   change_current_directory
4097 00012F63 0F8274FFFFFF              <1>             jc     loc_swp_retn ; eax = error code
4098                                    <1>
4099                                    <1> loc_swp_change_prompt_dir_string:
4100                                    <1>             ; esi = PATH_Array
4101                                    <1>             ; eax = Current Directory First Cluster
4102                                    <1>             ; edi = Logical DOS Drive Description Table
4103 00012F69 E8C37EFFFF                <1>             call   change_prompt_dir_str
4104 00012F6E 29C0                      <1>             sub    eax, eax ; 0
4105 00012F70 E968FFFFFF                <1>             jmp    loc_swp_retn
4106                                    <1>
4107                                    <1> reset_working_path:
4108                                    <1>             ; 06/10/2016 - TRDOS 386 (TRDOS v2.0)
4109                                    <1>             ;
4110                                    <1>             ; TRDOS v1.0 (DIR.ASM, "proc_reset_working_path")
4111                                    <1>             ; 05/02/2011 - 08/02/2011
4112                                    <1>             ;
4113                                    <1>             ; Restores current drive and directory
4114                                    <1>             ;
```

```
                                <1>             ; INPUT: none
                                <1>             ; OUTPUT: DL = SWP_DRV, EAX = 0 -> OK
                                <1>             ;
                                <1>             ;    AX = 0 -> ESI = Logical Dos Drv Desc. Table
                                <1>             ;
                                <1>             ;    EAX, EBX, ECX, EDX, ESI, EDI will be changed
                                <1>             ;
                                <1>
                                <1>
00012F75 31C0                   <1>             xor    eax, eax
00012F77 48                     <1>             dec    eax
                                <1>
00012F78 668B15[AA950100]       <1>             mov    dx, [SWP_DRV]
00012F7F 08F6                   <1>             or     dh, dh
00012F81 742E                   <1>             jz     short loc_rwp_return
                                <1>
00012F83 3A15[B6890100]         <1>             cmp    dl, [Current_Drv]
00012F89 7407                   <1>             je     short loc_rwp_restore_cdir
                                <1> loc_rwp_restore_cdrv:
00012F8B E81E50FFFF             <1>             call   change_current_drive
00012F90 EB10                   <1>             jmp    short loc_rwp_restore_ok
                                <1> loc_rwp_restore_cdir:
00012F92 31DB                   <1>             xor    ebx, ebx
00012F94 88D7                   <1>             mov    bh, dl
00012F96 BE00010900             <1>             mov    esi, Logical_DOSDisks
00012F9B 01DE                   <1>             add    esi, ebx
                                <1>
00012F9D E8C350FFFF             <1>             call   restore_current_directory
                                <1>
                                <1> loc_rwp_restore_ok:
00012FA2 668B15[AA950100]       <1>             mov    dx, [SWP_DRV]
00012FA9 31C0                   <1>             xor    eax, eax
00012FAB 66A3[AB950100]         <1>             mov    [SWP_DRV_chg], ax
                                <1> loc_rwp_return:
00012FB1 C3                     <1>             retn
                                <1>
                                <1> get_file_name:
                                <1>             ; 15/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>             ; Convert file name
                                <1>             ;      from directory entry format
                                <1>             ;   ;  to (8.3) dot file name format
                                <1>             ;
                                <1>             ; TRDOS v1.0 (DIR.ASM, "get_file_name")
                                <1>             ;    ; 2005 - 09/10/2011
                                <1>             ; INPUT:
                                <1>             ;      DS:SI -> Directory Entry Format File Name
                                <1>             ;        ES:DI -> DOS Dot File Name Address
                                <1>             ; OUTPUT:
                                <1>             ;      DS:SI -> DOS Dot File Name Address
                                <1>             ;   ;  ES:DI -> Directory Entry Format File Name
                                <1>             ;
                                <1>             ; TRDOS 386 (15/10/2016)
                                <1>             ; INPUT:
                                <1>             ;      ESI = File name addr in dir entry format
                                <1>             ;      EDI = Dot file name address (destination)
                                <1>             ; OUTPUT:
                                <1>             ;      File name is converted and moved
                                <1>             ;      to destination (as 8.3 dot filename)
                                <1>             ;
                                <1>             ; Modified registers: EAX, ECX
                                <1>
                                <1>                ; 2005 (TRDOS 8086) - 2016 (TRDOS 386)
                                <1>
00012FB2 57                     <1>             push   edi
00012FB3 56                     <1>             push   esi
00012FB4 AC                     <1>             lodsb
00012FB5 3C20                   <1>             cmp    al, 20h
00012FB7 762A                   <1>             jna    short pass_gfn_ext
00012FB9 56                     <1>             push   esi
00012FBA AA                     <1>             stosb
00012FBB B907000000             <1>             mov    ecx, 7
                                <1> loc_gfn_next_char:
00012FC0 AC                     <1>             lodsb
00012FC1 3C20                   <1>             cmp    al, 20h
00012FC3 7603                   <1>             jna    short pass_gfn_fn
00012FC5 AA                     <1>             stosb
00012FC6 E2F8                   <1>             loop   loc_gfn_next_char
                                <1> pass_gfn_fn:
00012FC8 5E                     <1>             pop    esi
00012FC9 83C607                 <1>             add    esi, 7
00012FCC AC                     <1>             lodsb
00012FCD 3C20                   <1>             cmp    al, 20h
00012FCF 7612                   <1>             jna    short pass_gfn_ext
00012FD1 B42E                   <1>             mov    ah, '.'
00012FD3 86E0                   <1>             xchg   ah, al
00012FD5 66AB                   <1>             stosw
00012FD7 AC                     <1>             lodsb
00012FD8 3C20                   <1>             cmp    al, 20h
00012FDA 7607                   <1>             jna    short pass_gfn_ext
00012FDC AA                     <1>             stosb
00012FDD AC                     <1>             lodsb
00012FDE 3C20                   <1>             cmp    al, 20h
00012FE0 7601                   <1>             jna    short pass_gfn_ext
00012FE2 AA                     <1>             stosb
                                <1> pass_gfn_ext:
00012FE3 30C0                   <1>             xor    al, al
00012FE5 AA                     <1>             stosb
00012FE6 5E                     <1>             pop    esi
00012FE7 5F                     <1>             pop    edi
00012FE8 C3                     <1>             retn
                                <1>
                                <1> set_hardware_int_vector:
                                <1>             ; 18/03/2017
                                <1>             ; 03/03/2017
                                <1>             ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
```

```
                                <1>             ;
                                <1>             ; SET/RESET HARDWARE INTERRUPT GATE
                                <1>             ;
                                <1>             ; Changes interrupt gate descriptor table
                                <1>             ; (without changing default interrupt list)
                                <1>             ;
                                <1>             ; INPUT:
                                <1>             ;      AL = IRQ number (0 to 15)
                                <1>             ;      AH > 0 -> set
                                <1>             ;      AH = 0 -> reset
                                <1>             ;
                                <1>             ; Modified registers: eax, ebx, edx, edi
                                <1>             ;
                                <1>
00012FE9 C0E002                 <1>             shl    al, 2 ; IRQ number * 4
00012FEC 0FB6D8                 <1>             movzx  ebx, al
                                <1>
00012FEF 08E4                   <1>             or     ah, ah
00012FF1 7508                   <1>             jnz    short shintv_1 ; set (for user call service)
                                <1>
                                <1>             ; 18/03/2017
00012FF3 81C3[80490100]         <1>             add    ebx, IRQ_list ; reset to default interrupt list
00012FF9 EB06                   <1>             jmp    short shintv_2
                                <1> shintv_1:
00012FFB 81C3[22300100]         <1>             add    ebx, IRQ_u_list
                                <1> shintv_2:
00013001 8B13                   <1>             mov    edx, [ebx] ; IRQ handler address
                                <1>
                                <1>             ; 03/03/2017
00013003 D0E0                   <1>             shl    al, 1 ; IRQ number * 8
                                <1>             ; 18/03/2017
00013005 0FB6F8                 <1>             movzx  edi, al
00013008 81C7[08870100]         <1>             add    edi, idt + (8*32) ; IRQ 0 offset = idt + 256
                                <1>
0001300E 89D0                   <1>             mov    eax, edx ; IRQ handler address
00013010 BB00000800             <1>             mov    ebx, 80000h
                                <1>
                                <1>             ;mov   edx, eax
00013015 66BA008E               <1>             mov    dx, 8E00h
00013019 6689C3                 <1>             mov    bx, ax
0001301C 89D8                   <1>             mov    eax, ebx ; /* selector = 0x0008 = cs */
                                <1>                                     ; /* interrupt gate - dpl=0, present */
0001301E AB                     <1>             stosd  ; selector & offset bits 0-15
0001301F 8917                   <1>             mov    [edi], edx ; attributes & offset bits 16-23
                                <1>
00013021 C3                     <1>             retn
                                <1> IRQ_u_list:
                                <1>             ; 28/02/2017
00013022 [54090000]             <1>             dd     timer_int
00013026 [C6100000]             <1>             dd     kb_int
0001302A [360B0000]             <1>             dd     irq2
0001302E [62300100]             <1>             dd     IRQ_service3
00013032 [6C300100]             <1>             dd     IRQ_service4
00013036 [76300100]             <1>             dd     IRQ_service5
0001303A [F3500000]             <1>             dd     fdc_int
0001303E [80300100]             <1>             dd     IRQ_service7
00013042 [BF0A0000]             <1>             dd     rtc_int
00013046 [8A300100]             <1>             dd     IRQ_service9
0001304A [94300100]             <1>             dd     IRQ_service10
0001304E [9E300100]             <1>             dd     IRQ_service11
00013052 [A8300100]             <1>             dd     IRQ_service12
00013056 [B2300100]             <1>             dd     IRQ_service13
0001305A [605A0000]             <1>             dd     hdc1_int
0001305E [835A0000]             <1>             dd     hdc2_int
                                <1>
                                <1>             ; 03/03/2017
                                <1>             ; 27/02/2017
                                <1> IRQ_service3:
00013062 36C605[6E9B0100]03     <1>             mov    byte [ss:IRQnum], 3
0001306A EB4E                   <1>             jmp    short IRQ_service
                                <1> IRQ_service4:
0001306C 36C605[6E9B0100]04     <1>             mov    byte [ss:IRQnum], 4
00013074 EB44                   <1>             jmp    short IRQ_service
                                <1> IRQ_service5:
00013076 36C605[6E9B0100]05     <1>             mov    byte [ss:IRQnum], 5
0001307E EB3A                   <1>             jmp    short IRQ_service
                                <1> IRQ_service7:
00013080 36C605[6E9B0100]07     <1>             mov    byte [ss:IRQnum], 7
00013088 EB30                   <1>             jmp    short IRQ_service
                                <1> IRQ_service9:
0001308A 36C605[6E9B0100]09     <1>             mov    byte [ss:IRQnum], 9
00013092 EB26                   <1>             jmp    short IRQ_service
                                <1> IRQ_service10:
00013094 36C605[6E9B0100]0A     <1>             mov    byte [ss:IRQnum], 10
0001309C EB1C                   <1>             jmp    short IRQ_service
                                <1> IRQ_service11:
0001309E 36C605[6E9B0100]0B     <1>             mov    byte [ss:IRQnum], 11
000130A6 EB12                   <1>             jmp    short IRQ_service
                                <1> IRQ_service12:
000130A8 36C605[6E9B0100]0C     <1>             mov    byte [ss:IRQnum], 12
000130B0 EB08                   <1>             jmp    short IRQ_service
                                <1> IRQ_service13:
000130B2 36C605[6E9B0100]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
000130BA 1E                     <1>             push   ds
000130BB 06                     <1>             push   es
000130BC 0FA0                   <1>             push   fs
000130BE 0FA8                   <1>             push   gs
                                <1>
```

```
4325 000130C0 60                  <1>             pushad ; eax,ecx,edx,ebx,esp,ebp,esi,edi
4326 000130C1 66B91000            <1>             mov     cx, KDATA
4327 000130C5 8ED9                <1>             mov     ds, cx
4328 000130C7 8EC1                <1>             mov     es, cx
4329 000130C9 8EE1                <1>             mov     fs, cx
4330 000130CB 8EE9                <1>             mov     gs, cx
4331                              <1>
4332 000130CD 0F20D8              <1>             mov    eax, cr3
4333 000130D0 A3[6A9B0100]        <1>             mov    [IRQ_cr3], eax
4334                              <1>
4335 000130D5 A1[F0880100]        <1>             mov    eax, [k_page_dir]
4336 000130DA 0F22D8              <1>             mov    cr3, eax
4337                              <1>
4338 000130DD A0[6E9B0100]        <1>             mov    al, [IRQnum]
4339                              <1>
4340                              <1>             ;mov   cl, [sysflg]
4341                              <1>             ;mov   [u.r_mode], cl  ; system (0) or user mode (FFh)
4342                              <1> IRQsrv_0:
4343 000130E2 0FB6D8              <1>             movzx  ebx, al
4344 000130E5 8A9B[B8480100]      <1>             mov    bl, [ebx+IRQenum] ; IRQ (available) index number + 1
4345                              <1>             ; 01/03/2017
4346 000130EB FECB                <1>             dec    bl  ; IRQ index number, 0 to 8
4347 000130ED 0F8807010000        <1>             js     IRQsrv_5 ; not available to use here!?
4348                              <1>             ;
4349 000130F3 80BB[349B0100]80    <1>             cmp    byte [ebx+IRQ.method], 80h ; using by a dev or kernel?
4350 000130FA 7205                <1>             jb     short IRQsrv_1 ; no
4351                              <1>
4352                              <1>             ; If the IRQ service is already owned by TRDOS 386 kernel
4353                              <1>             ;       or a Device driver
4354                              <1>             ; we need to call 'dev_IRQ_service'
4355                              <1>
4356                              <1>             ; IRQ number in AL
4357 000130FC E866020000          <1>             call   dev_IRQ_service      ; IRQ service for device drivers
4358                              <1>             ; IRQ number in AL
4359                              <1> IRQsrv_1:
4360                              <1>             ; check user callback service status
4361                              <1>             ; AL = IRQ number
4362                              <1>             ; EBX = IRQ (Available) Index number
4363                              <1>
4364 00013101 A2[D7030300]        <1>             mov    [u.irqwait], al ; set waiting IRQ flag
4365                              <1>
4366 00013106 8A83[229B0100]      <1>             mov    al, [ebx+IRQ.owner]
4367 0001310C 20C0                <1>             and    al, al
4368 0001310E 0F84E6000000        <1>             jz     IRQsrv_5 ; it is not owned by a user/proc
4369                              <1>
4370                              <1>             ; 03/03/2017
4371 00013114 89DA                <1>             mov    edx, ebx
4372 00013116 C0E202              <1>             shl    dl, 2
4373 00013119 8B92[469B0100]      <1>             mov    edx, [edx+IRQ.addr] ; S.R.B. or Callback service addr
4374                              <1>
4375 0001311F 8AA3[349B0100]      <1>             mov    ah, [ebx+IRQ.method]
4376 00013125 F6C401              <1>             test   ah, 1
4377 00013128 7534                <1>             jnz    short IRQsrv_4 ; Callback service method
4378                              <1>
4379                              <1>             ; Signal Response Byte method
4380                              <1>             ;mov   edx, [edx+IRQ.addr] ; Signal Response Byte address
4381                              <1>             ;                          ; (Physical address, non-swappable)
4382 0001312A 80E402              <1>             and    ah, 2 ; bit 1, (S.R.B.) counter (auto increment) method
4383 0001312D 8AA3[3D9B0100]      <1>             mov    ah, [ebx+IRQ.srb] ; Signal Response Byte value
4384 00013133 7408                <1>             jz     short IRQsrv_2 ; fixed S.R.B. value
4385                              <1>             ; counter method (auto increment)
4386 00013135 FEC4                <1>             inc    ah
4387 00013137 88A3[3D9B0100]      <1>             mov    [ebx+IRQ.srb], ah ; next (count) number
4388                              <1> IRQsrv_2:
4389 0001313D 8822                <1>             mov    [edx], ah ; put S.R.B. val to the user's S.R.B. addr
4390 0001313F C605[D7030300]00    <1>             mov    byte [u.irqwait], 0 ; clear waiting IRQ flag
4391                              <1>
4392 00013146 3A05[B3030300]      <1>             cmp    al, [u.uno]
4393 0001314C 0F84A8000000        <1>             je     IRQsrv_5 ; the owner is current user/process
4394                              <1> IRQsrv_3:
4395                              <1>             ; the owner is not current user/process
4396                              <1>             ; AL = process number
4397 00013152 B202                <1>             mov    dl, 2 ; priority, 2 = event (high)
4398 00013154 E837FAFFFF          <1>             call   set_run_sequence
4399                              <1>
4400                              <1>             ; [u.irqwait] = waiting IRQ number for callback service
4401                              <1>
4402 00013159 E99C000000          <1>             jmp    IRQsrv_5
4403                              <1> IRQsrv_4:
4404 0001315E 3A05[B3030300]      <1>             cmp    al, [u.uno]  ; is the owner is current user/process?
4405 00013164 75EC                <1>             jne    short IRQsrv_3 ; no !
4406                              <1>
4407                              <1>             ; Check if an IRQ callback service already in progress
4408 00013166 803D[D8030300]00    <1>             cmp    byte [u.r_lock], 0
4409 0001316D 0F8787000000        <1>             ja     IRQsrv_5 ; nothing to do !
4410                              <1>                             ; (we need to complete prev callback)
4411 00013173 803D[D4030300]00    <1>             cmp    byte [u.t_lock], 0
4412 0001317A 777E                <1>             ja     short IRQsrv_5 ; nothing to do !
4413                              <1>                             ; (we need to complete timer callback)
4414                              <1>
4415                              <1>             ; 04/03/2017
4416 0001317C C605[D7030300]00    <1>             mov    byte [u.irqwait], 0 ; reset/clear waiting IRQ flag
4417                              <1>
4418 00013183 FE05[D8030300]      <1>             inc    byte [u.r_lock] ; 'IRQ callback service in progress' flag
4419                              <1>
4420 00013189 8A0D[5B030300]      <1>             mov    cl, [sysflg]   ; (system call) mode flag (kernel/user)
4421 0001318F 880D[D9030300]      <1>             mov    [u.r_mode], cl ; system mode (0) or user mode (FFh)
4422                              <1>
4423                              <1>             ;
4424 00013195 8B2D[8C880100]      <1>             mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
4425 0001319B 83ED14              <1>             sub    ebp, 20             ; eip, cs, eflags, esp, ss
4426 0001319E 892D[5C030300]      <1>             mov    [u.sp], ebp
4427 000131A4 8925[60030300]      <1>             mov    [u.usp], esp
4428                              <1>
4429                              <1>             ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
```

```
4430                                    <1>
4431 000131AA 8B44241C                  <1>             mov     eax, [esp+28] ; pushed eax
4432 000131AE A3[64030300]              <1>             mov     [u.r0], eax
4433                                    <1>
4434 000131B3 E820E7FFFF                <1>             call    wswap ; save user's registers & status
4435                                    <1>
4436                                    <1>             ; software int is in ring 0 but IRQ handler must return to ring 3
4437                                    <1>             ; so, ring 3 return address and stack registers
4438                                    <1>             ; (eip, cs, eflags, esp, ss)
4439                                    <1>             ; must be copied to IRQ handler return
4440                                    <1>             ; eip will be replaced by callback service routine address
4441                                    <1>
4442 000131B8 C605[5B030300]FF          <1>             mov     byte [sysflg], 0FFh ; user mode
4443                                    <1>
4444                                    <1>             ; system mode (system call)
4445                                    <1>             ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
4446                                    <1>                                ; ESP (u), SS (UDATA)
4447                                    <1>
4448 000131BF 8B4510                   <1>             mov     eax, [ebp+16]; SS (UDATA)
4449 000131C2 89E6                     <1>             mov     esi, esp
4450 000131C4 50                       <1>             push    eax
4451 000131C5 50                       <1>             push    eax
4452 000131C6 89E7                     <1>             mov     edi, esp
4453 000131C8 893D[60030300]           <1>             mov     [u.usp], edi
4454 000131CE B908000000               <1>             mov     ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
4455 000131D3 F3A5                     <1>             rep     movsd
4456 000131D5 B104                     <1>             mov     cl, 4
4457 000131D7 F3AB                     <1>             rep     stosd
4458 000131D9 893D[5C030300]           <1>             mov     [u.sp], edi
4459 000131DF 89EE                     <1>             mov     esi, ebp
4460 000131E1 B105                     <1>             mov     cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
4461 000131E3 F3A5                     <1>             rep     movsd
4462                                    <1>             ;
4463                                    <1>
4464 000131E5 8B0D[B8030300]           <1>             mov     ecx, [u.pgdir]
4465 000131EB 890D[6A9B0100]           <1>             mov     [IRQ_cr3], ecx
4466                                    <1>
4467                                    <1> set_IRQ_callback_addr:
4468                                    <1>             ;
4469                                    <1>             ; This routine sets return address
4470                                    <1>             ; to start of user's interrupt
4471                                    <1>             ; service (callback) address
4472                                    <1>             ;
4473                                    <1>             ; INPUT:
4474                                    <1>             ;     EDX = callback routine/service address
4475                                    <1>             ;           (virtual, not physical address!)
4476                                    <1>             ;     [u.sp] = kernel stack, points to
4477                                    <1>             ;              user's EIP,CS,EFLAGS,ESP,SS
4478                                    <1>             ;              registers.
4479                                    <1>             ; OUTPUT:
4480                                    <1>             ;     EIP (user) = callback (service) address
4481                                    <1>             ;     CS (user) = UCODE
4482                                    <1>             ;     EFLAGS (user) = flags before callback
4483                                    <1>             ;      ESP (user) = ESP-4 (user, before callback)
4484                                    <1>             ;     [ESP](user) = EIP (user) before callback
4485                                    <1>             ;
4486                                    <1>             ; Note: If CPU was in user mode while entering
4487                                    <1>             ;     the timer interrupt service routine,
4488                                    <1>             ;     'IRET' will get return to callback routine
4489                                    <1>             ;     immediately. If CPU was in system/kernel mode
4490                                    <1>             ;     'iret' will get return to system call and
4491                                    <1>             ;     then, callback routine will be return address
4492                                    <1>             ;     from system call. (User's callback/service code
4493                                    <1>             ;     will be able to return to normal return address
4494                                    <1>             ;     via a 'sysrele' system call at the end.)
4495                                    <1>             ;
4496                                    <1>             ; Note: User's IRQ callback service code must be ended
4497                                    <1>             ;     with a 'sysrele' system call !
4498                                    <1>             ;
4499                                    <1>             ;     For example:
4500                                    <1>             ;
4501                                    <1>             ;     audio IRQ callback:
4502                                    <1>             ;         ...
4503                                    <1>             ;         <load DMA buffer with audio data>
4504                                    <1>             ;         ...
4505                                    <1>             ;         mov eax, 39 ; 'sysrele'
4506                                    <1>             ;         int 40h ; TRDOS 386 system call (interrupt)
4507                                    <1>             ;
4508                                    <1>
4509                                    <1>             ;mov   edx, [edx+IRQ.addr] ; Callback service address
4510                                    <1>             ;                       ; (Virtual address)
4511                                    <1>
4512 000131F1 8B2D[5C030300]           <1>             mov     ebp, [u.sp]; kernel's stack, points to EIP (user)
4513 000131F7 895500                   <1>             mov     [ebp], edx
4514                                    <1> IRQsrv_5:
4515                                    <1>             ; EOI & return
4516                                    <1>             ; 01/08/2020
4517                                    <1>             ; 11/06/2017
4518                                    <1>             ; 10/06/2017
4519                                    <1>             ;mov   al, [IRQnum]
4520 000131FA B020                     <1>             mov     al, 20h ; 01/08/2020
4521 000131FC FA                       <1>             cli
4522                                    <1>             ;cmp   al, 7
4523 000131FD 803D[6E9B0100]07         <1>             cmp     byte [IRQnum], 7 ; 01/08/2020
4524 00013204 7602                     <1>             jna     short IRQsrv_6
4525                                    <1>             ;
4526                                    <1>             ;;mov  al, EOI      ; end of interrupt
4527                                    <1>             ;mov   al, 20h ; 01/08/2020
4528                                    <1>             ;cli         ; disable interrupts till stack cleared
4529                                    <1>             ;out   INTB00, al ; For controll2 #2
4530 00013206 E6A0                     <1>             out     0A0h, al
4531                                    <1> IRQsrv_6:
4532                                    <1>             ;mov   byte [IRQnum], 0 ; reset
4533                                    <1>             ;;mov  al, EOI      ; end of interrupt
4534                                    <1>             ;mov   al, 20h ; 01/08/2020
```

```
                                <1>             ;cli         ; disable interrupts till stack cleared
                                <1>             ;out   INTA00, al ; end of interrupt to 8259 - 1
00013208 E620                   <1>             out    20h, al
                                <1> IRQsrv_7:
                                <1>             ;; 13/06/2017
                                <1>             ;or    word [ebp+8], 200h ; force enabling interrupts
                                <1>             ;
0001320A 8B0D[6A9B0100]         <1>             mov    ecx, [IRQ_cr3]      ; previous content of cr3 register
00013210 0F22D9                 <1>             mov    cr3, ecx     ; restore cr3 register content
                                <1>             ;
00013213 61                     <1>             popad ; edi,esi,ebp,(icrement esp by 4),ebx,edx,ecx,eax
                                <1>             ;
00013214 0FA9                   <1>             pop    gs
00013216 0FA1                   <1>             pop    fs
00013218 07                     <1>             pop    es
00013219 1F                     <1>             pop    ds
                                <1>             ;
0001321A CF                     <1>             iretd  ; return from interrupt
                                <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>
0001321B BF[AD950100]           <1>             mov    edi, device_name
00013220 E805010000             <1>             call   lodsb_capitalize
00013225 88C4                   <1>             mov    ah, al
00013227 3C2F                   <1>             cmp    al, '/'
00013229 750E                   <1>             jne    short gdn_1
0001322B BF[AD950100]           <1>             mov    edi, device_name
00013230 E8F5000000             <1>             call   lodsb_capitalize
                                <1> gdn_0:
00013235 20C0                   <1>             and    al, al ; 0 ?
00013237 7420                   <1>             jz     short gdn_err ; null name after '/'
                                <1> gdn_1:
00013239 3C44                   <1>             cmp    al, 'D'
0001323B 7517                   <1>             jne    short gdn_2
0001323D E8E8000000             <1>             call   lodsb_capitalize
00013242 3C45                   <1>             cmp    al, 'E'
00013244 750E                   <1>             jne    short gdn_2
00013246 E8DF000000             <1>             call   lodsb_capitalize
0001324B 3C56                   <1>             cmp    al, 'V'
0001324D 7505                   <1>             jne    short gdn_2
0001324F AC                     <1>             lodsb
00013250 3C2F                   <1>             cmp    al, '/'
00013252 740D                   <1>             je     short gdn_4
                                <1> gdn_2:
00013254 80FC2F                 <1>             cmp    ah, '/'
00013257 750F                   <1>             jne    short gdn_5
                                <1> gdn_err:
                                <1>             ; invalid device name or device not found
00013259 31C0                   <1>             xor    eax, eax ; 0
0001325B F9                     <1>             stc
0001325C C3                     <1>             retn
                                <1> gdn_3:
0001325D 3C2F                   <1>             cmp    al, '/'
0001325F 7507                   <1>             jne    short gdn_5
                                <1> gdn_4:
00013261 BF[AD950100]           <1>             mov    edi, device_name
00013266 EB04                   <1>             jmp    short gdn_6
                                <1> gdn_5:
00013268 3C00                   <1>             cmp    al, 0
0001326A 7419                   <1>             je     short gdn_7
                                <1> gdn_6:
0001326C E8B9000000             <1>             call lodsb_capitalize
00013271 81FF[B5950100]         <1>             cmp    edi, device_name + 8
00013277 72E4                   <1>             jb     short gdn_3
00013279 3C00                   <1>             cmp    al, 0
0001327B 75DC                   <1>             jne    short gdn_err
0001327D 81FF[AE950100]         <1>             cmp    edi, device_name + 1
00013283 76D4                   <1>             jna    short gdn_err ; null name after '/'
                                <1> gdn_7:
00013285 AA                     <1>             stosb
                                <1>             ; zero padding ("NAME",0,0,0,0)
00013286 81FF[B5950100]         <1>             cmp    edi, device_name + 8
0001328C 72F7                   <1>             jb     short gdn_7
                                <1> gdn_8:
```

```
4640                                    <1>             ; search for kernel device names
4641 0001328E BE[AD950100]              <1>             mov    esi, device_name
4642 00013293 BF[9E460100]              <1>             mov    edi, KDEV_NAME
4643 00013298 31C0                      <1>             xor    eax, eax
4644                                    <1> gdn_9:
4645 0001329A A7                        <1>             cmpsd
4646 0001329B 7505                      <1>             jne    short gdn_10
4647 0001329D A7                        <1>             cmpsd
4648 0001329E 7503                      <1>             jne    short gdn_11
4649 000132A0 EB2B                      <1>             jmp    short gdn_17 ; match
4650                                    <1> gdn_10:
4651 000132A2 A7                        <1>             cmpsd  ; add esi, 4 & add edi, 4
4652                                    <1> gdn_11:
4653 000132A3 BE[AD950100]              <1>             mov    esi, device_name
4654 000132A8 FEC0                      <1>             inc    al
4655 000132AA 3C16                      <1>             cmp    al, NumOfKernelDevNames
4656 000132AC 72EC                      <1>             jb     short gdn_9
4657                                    <1> gdn_12:
4658                                    <1>             ; search for installable device names
4659                                    <1>             ; esi = offset device_name
4660 000132AE BF[D8950100]              <1>             mov    edi, IDEV_NAME
4661 000132B3 28C0                      <1>             sub    al, al ; 0
4662                                    <1> gdn_13:
4663 000132B5 A7                        <1>             cmpsd
4664 000132B6 7505                      <1>             jne    short gdn_14
4665 000132B8 A7                        <1>             cmpsd
4666 000132B9 7503                      <1>             jne    short gdn_15
4667 000132BB EB3F                      <1>             jmp    short gdn_19 ; match
4668                                    <1> gdn_14:
4669 000132BD A7                        <1>             cmpsd  ; add esi, 4 & add edi, 4
4670                                    <1> gdn_15:
4671 000132BE BE[AD950100]              <1>             mov    esi, device_name
4672 000132C3 FEC0                      <1>             inc    al
4673 000132C5 3C08                      <1>             cmp    al, NumOfInstallableDevices
4674 000132C7 72EC                      <1>             jb     short gdn_13
4675                                    <1>
4676                                    <1> gdn_16:     ; error: invalid device name (not found) !
4677 000132C9 30C0                      <1>             xor    al, al
4678 000132CB F9                        <1>             stc
4679 000132CC C3                        <1>             retn
4680                                    <1>
4681                                    <1> gdn_17:     ; name match (with one of kernel device names)
4682                                    <1>             ;
4683                                    <1>             ; convert KDEV_NAME index to
4684                                    <1>             ; KDEV_NUMBER index
4685                                    <1>             ; (different names are used for same devices)
4686                                    <1>             ; (example: "COM1" & "TTY8" = device number 18)
4687 000132CD 89C3                      <1>             mov    ebx, eax ; < 256
4688 000132CF 8A83[4E470100]            <1>             mov    al, [KDEV_NUMBER+ebx]
4689                                    <1>
4690                                    <1>             ; check if empty dev entry in the list
4691 000132D5 80B8[5C970100]00          <1>             cmp    byte [DEV_OPENMODE+eax], 0
4692 000132DC 771B                      <1>             ja     short gdn_18 ; it must be already set
4693                                    <1>
4694                                    <1>             ; (re)set device name and access flags
4695                                    <1>             ; (remain open work will be easy after that)
4696                                    <1>             ; (NOTE: here, data will be copied to bss section)
4697 000132DE 88C3                      <1>             mov    bl, al
4698 000132E0 83EF08                    <1>             sub    edi, 8 ; kernel device name address (data)
4699 000132E3 66C1E302                  <1>             shl    bx, 2
4700 000132E7 89BB[7A970100]            <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
4701 000132ED 8A98[A4480100]            <1>             mov    bl, [KDEV_ACCESS+eax] ; kernel dev list (data)
4702 000132F3 8898[A8960100]            <1>             mov    [DEV_ACCESS+eax], bl ; (all) device list (bss)
4703                                    <1> gdn_18:
4704 000132F9 FEC0                      <1>             inc    al ; 1 to NumOfKernelDevNames (<=7Fh)
4705                                    <1>             ; eax = device index/entry number
4706 000132FB C3                        <1>             retn
4707                                    <1>
4708                                    <1> gdn_19:     ; name match (with one of installable device names)
4709                                    <1>             ;
4710                                    <1>             ; al = 0 to NumOfInstallableDevices - 1 (<=7Fh)
4711                                    <1>
4712 000132FC 89C3                      <1>             mov    ebx, eax
4713 000132FE 80C316                    <1>             add    bl, NumOfKernelDevices    ; < NUMOFDEVICES
4714                                    <1>
4715                                    <1>             ; check if empty dev entry in the list
4716 00013301 80BB[5C970100]00          <1>             cmp    byte [DEV_OPENMODE+ebx], 0
4717 00013308 771D                      <1>             ja     short gdn_20 ; 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 0001330A 83EF08                    <1>             sub    edi, 8 ; installable device name address
4722 0001330D 66C1E302                  <1>             shl    bx, 2 ;*4
4723 00013311 89BB[7A970100]            <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
4724 00013317 66C1EB02                  <1>             shr    bx, 2
4725 0001331B 8A80[20960100]            <1>             mov    al, [IDEV_FLAGS+eax] ; installable dev list
4726 00013321 8883[A8960100]            <1>             mov    [DEV_ACCESS+ebx], al ; (all) device list
4727                                    <1> gdn_20:
4728 00013327 88D8                      <1>             mov    al, bl
4729                                    <1>             ; eax = device index/entry number ; < NUMOFDEVICES
4730 00013329 C3                        <1>             retn
4731                                    <1>
4732                                    <1> lodsb_capitalize:
4733                                    <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
4734                                    <1>       ; INPUT -> [esi] = character
4735                                    <1>       ;          edi = destination
4736                                    <1>       ; OUTPUT -> AL contains capitalized character
4737                                    <1>       ;         esi = esi+1
4738                                    <1>       ;         edi = edi+1
4739                                    <1>       ;
4740 0001332A AC                        <1>       lodsb
4741 0001332B 3C61                      <1>       cmp al, 61h
4742 0001332D 7206                      <1>       jb  short lodsb_cap_retn
4743 0001332F 3C7A                      <1>       cmp al, 7Ah
4744 00013331 7702                      <1>       ja  short lodsb_cap_retn
```

```
4745 00013333 24DF              <1>       and al, 0DFh
4746                            <1> lodsb_cap_retn:
4747 00013335 AA                <1>       stosb
4748 00013336 C3                <1>       retn
4749                            <1>
4750                            <1> device_open:
4751                            <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
4752                            <1>       ; Complete device opening work for sysopen (device)
4753                            <1>       ;
4754                            <1>       ; INPUT ->
4755                            <1>       ;      EAX = Device Number (AL)
4756                            <1>       ;        CL = Open mode (1 = read, 2 = write)
4757                            <1>       ;       CH = Device access byte (bit 0 = 0)
4758                            <1>       ; OUTPUT ->
4759                            <1>       ;      EAX = Device Number
4760                            <1>       ;      CF = 0 -> device has been opened
4761                            <1>       ;      CF = 1 -> device could not be opened
4762                            <1>       ;
4763                            <1>       ;  Modified registers: ebx, (edx, ecx, esi, edi, ebp)
4764                            <1>       ;
4765                            <1>
4766 00013337 89C3              <1>       mov    ebx, eax
4767 00013339 66C1E302          <1>       shl    bx, 2 ; *4
4768                            <1>
4769 0001333D F6C580            <1>       test   ch, 80h ; bit 7, installable device driver flag
4770 00013340 7406              <1>       jz     short d_open_2 ; Kernel device
4771                            <1>       ; installable device
4772                            <1> d_open_1:
4773 00013342 FFA3[24960100]    <1>         jmp  dword [ebx+IDEV_OADDR-4]
4774                            <1> d_open_2:
4775 00013348 FFA3[60470100]    <1>       jmp    dword [ebx+KDEV_OADDR-4]
4776                            <1>
4777                            <1> device_close:
4778                            <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
4779                            <1>       ; Complete device closing work for sysclose (device)
4780                            <1>       ;
4781                            <1>       ; INPUT ->
4782                            <1>       ;      EAX = Device Number (AL)
4783                            <1>       ;        CL = Open mode (1 = read, 2 = write)
4784                            <1>       ;       CH = Device access byte (bit 0 = 0)
4785                            <1>       ; OUTPUT ->
4786                            <1>       ;      EAX = Device Number
4787                            <1>       ;      CF = 0 -> device has been closed
4788                            <1>       ;      CF = 1 -> device could not be closed
4789                            <1>       ;
4790                            <1>       ; Modified registers: ebx, (edx, ecx, esi, edi, ebp)
4791                            <1>       ;
4792                            <1>
4793 0001334E 89C3              <1>       mov    ebx, eax
4794 00013350 66C1E302          <1>       shl    bx, 2 ; *4
4795                            <1>
4796 00013354 F6C580            <1>       test   ch, 80h ; bit 7, installable device driver flag
4797 00013357 7406              <1>       jz     short d_close_2 ; Kernel device
4798                            <1>       ; installable device
4799                            <1> d_close_1:
4800 00013359 FFA3[44960100]    <1>         jmp  dword [ebx+IDEV_CADDR-4]
4801                            <1> d_close_2:
4802 0001335F FFA3[B0470100]    <1>       jmp    dword [ebx+KDEV_CADDR-4]
4803                            <1>
4804                            <1> rnull:
4805                            <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
4806                            <1>       ; read null (read from null device)
4807 00013365 C3                <1>       retn
4808                            <1>
4809                            <1> wnull:
4810                            <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
4811                            <1>       ; write null (write to null device)
4812 00013366 C3                <1>       retn
4813                            <1>
4814                            <1> dev_IRQ_service:
4815                            <1>       ; 12/05/2017
4816                            <1>       ; 13/04/2017
4817                            <1>       ; 27/02/2017 - TRDOS 386 (TRDOS v2.0)
4818                            <1>       ; INPUT ->
4819                            <1>       ;      AL = IRQ Number (0 to 15)
4820                            <1>       ;
4821 00013367 53                <1>       push   ebx
4822 00013368 0FB6D8            <1>       movzx  ebx, al
4823 0001336B C0E302            <1>       shl    bl, 2 ; * 4
4824 0001336E 8B9B[E29A0100]    <1>       mov    ebx, [ebx+DEV_INT_HNDLR]
4825 00013374 21DB              <1>       and    ebx, ebx
4826 00013376 7404              <1>         jz   short dIRQ_s_retn
4827 00013378 50                <1>       push   eax
4828                            <1>
4829 00013379 FFD3              <1>       call   ebx
4830                            <1>
4831 0001337B 58                <1>       pop    eax
4832                            <1> dIRQ_s_retn:
4833 0001337C 5B                <1>       pop    ebx
4834 0001337D C3                <1>       retn
4835                            <1>
4836                            <1>
4837                            <1> set_dev_IRQ_service:
4838                            <1>       ; 13/04/2017 - TRDOS 386 (TRDOS v2.0)
4839                            <1>       ;
4840                            <1>       ; Set Device Interrupt Service
4841                            <1>       ;
4842                            <1>       ; INPUT ->
4843                            <1>       ;      AL = IRQ Number
4844                            <1>       ;      EBX = Hardware Interrupt Service Address
4845                            <1>       ;
4846                            <1>       ; Note: There is not a validation check here
4847                            <1>       ;      because this procedure is called by
4848                            <1>       ;      TRDOS 386 kernel !
4849                            <1>       ;        (Even if a device driver does not exist
```

```
4850                                    <1>       ;     this setting may be used by sysaudio
4851                                    <1>       ;     and other system calls for hardware
4852                                    <1>       ;     components which use IRQ method for I/O.)
4853                                    <1>       ;
4854                                    <1>       ;push  esi
4855 0001337E 0FB6F0                   <1>       movzx  esi, al
4856 00013381 66C1E602                 <1>       shl    si, 2 ; * 4
4857 00013385 899E[E29A0100]           <1>       mov    [esi+DEV_INT_HNDLR], ebx
4858                                    <1>       ;pop   esi
4859 0001338B C3                       <1>       retn
4860                                    <1>
4861                                    <1>
4862                                    <1> sysaudio: ; AUDIO FUNCTIONS
4863                                    <1>       ; 12/02/2021 (TRDOS 386 v2.0.3)
4864                                    <1>       ; 28/07/2020
4865                                    <1>       ; 27/07/2020
4866                                    <1>       ; 10/10/2017
4867                                    <1>       ; 22/06/2017
4868                                    <1>       ; 28/05/2017, 04/06/2017, 05/06/2017, 10/06/2017
4869                                    <1>       ; 01/05/2017, 12/05/2017, 15/05/2017, 20/05/2017
4870                                    <1>       ; 21/04/2017, 22/04/2017, 23/04/2017, 24/04/2017
4871                                    <1>       ; 10/04/2017, 13/04/2017, 14/04/2017, 16/04/2017
4872                                    <1>       ; 03/04/2017 (VIA VT8237R)
4873                                    <1>       ; 01/04/2016 (trdosk6.s -> tdosk8.s)
4874                                    <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
4875                                    <1>       ;
4876                                    <1>       ; Inputs:
4877                                    <1>       ;
4878                                    <1>       ;     BH = 0 -> Beep (PC Speaker)
4879                                    <1>       ;          BL = Duration Counter (1 for 1/64 second)
4880                                    <1>       ;          CX = Frequency Divisor (1193180/Frequency)
4881                                    <1>       ;             (1331 for 886 Hz)
4882                                    <1>       ;
4883                                    <1>       ;     01/04/2017
4884                                    <1>       ;
4885                                    <1>       ;     BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
4886                                    <1>       ;          BL = 0 : PC SPEAKER
4887                                    <1>       ;              1 : SOUND BLASTER 16
4888                                    <1>       ;              2 : INTEL AC'97
4889                                    <1>       ;              3 : VIA VT8237R (VT8233)
4890                                    <1>       ;              4 : INTEL HDA
4891                                    <1>       ;           5-FEh : unknown/invalid
4892                                    <1>       ;             ; 04/06/2017
4893                                    <1>       ;            FFh : Get current audio device id
4894                                    <1>       ;
4895                                    <1>       ;     BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
4896                                    <1>       ;           ECX = Audio Buffer Size (must be equal to
4897                                    <1>       ;                 the half of DMA buffer size)
4898                                    <1>       ;           EDX = Virtual Address of the buffer
4899                                    <1>       ;                 (This is not DMA buffer!)
4900                                    <1>       ;
4901                                    <1>       ;     BH = 3 -> INITIALIZE AUDIO DEVICE
4902                                    <1>       ;          BL = 0,2 -> for Signal Response Byte
4903                                    <1>       ;           CL = Signal Response Byte Value (fixed)
4904                                    <1>       ;                   if BL = 0
4905                                    <1>       ;                auto increment of S.R.B. value
4906                                    <1>       ;                   if BL = 2
4907                                    <1>       ;             EDX = Signal Response (Return) Byte Address
4908                                    <1>       ;
4909                                    <1>       ;          BL = 1 for CallBack Method
4910                                    <1>       ;           EDX = CallBack Service Address (Virtual)
4911                                    <1>       ;
4912                                    <1>       ;          BL > 2 -> invalid function
4913                                    <1>       ;
4914                                    <1>       ;         (Audio buffer must be allocated before
4915                                    <1>       ;          initialization.)
4916                                    <1>       ;
4917                                    <1>       ;     BH = 4 -> START TO PLAY
4918                                    <1>       ;          BL = Mode
4919                                    <1>       ;              Bit 0 = mono/stereo (1 = stereo)
4920                                    <1>       ;              Bit 1 = 8 bit / 16 bit (1 = 16 bit)
4921                                    <1>       ;          CX = Sampling Rate (Hz)
4922                                    <1>       ;
4923                                    <1>       ;     BH = 5 -> PAUSE
4924                                    <1>       ;          BL = Any
4925                                    <1>       ;
4926                                    <1>       ;     BH = 6 -> CONTINUE TO PLAY
4927                                    <1>       ;          BL = Any
4928                                    <1>       ;
4929                                    <1>       ;     BH = 7 -> STOP
4930                                    <1>       ;          BL = Any
4931                                    <1>       ;
4932                                    <1>       ;     BH = 8 -> RESET
4933                                    <1>       ;          BL = Any
4934                                    <1>       ;
4935                                    <1>       ;     BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
4936                                    <1>       ;          BL = Any
4937                                    <1>       ;
4938                                    <1>       ;     BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
4939                                    <1>       ;          BL = Any
4940                                    <1>       ;
4941                                    <1>       ;     BH = 11 -> SET VOLUME LEVEL
4942                                    <1>       ;          BL: (Bit 0 to 6)
4943                                    <1>       ;              0 = Master (Playback, Lineout) volume
4944                                    <1>       ;          CL = Left Channel Volume
4945                                    <1>       ;          CH = Right Channel Volume
4946                                    <1>       ;
4947                                    <1>       ;          Note: If BL >= 80h (Bit 7 of BL is set),
4948                                    <1>       ;          volume level will be set for next playing
4949                                    <1>       ;          (actual volume level will not be changed
4950                                    <1>       ;          immediately)
4951                                    <1>       ;
4952                                    <1>       ;     BH = 12 -> DISABLE AUDIO DEVICE
4953                                    <1>       ;          (reset audio device and unlink dma buffer)
4954                                    <1>       ;          BL = Any
```

```
4955                    <1>     ;
4956                    <1>     ;       12/05/2017
4957                    <1>     ;       BH = 13 -> MAP DMA BUFFER TO USER
4958                    <1>     ;           (for direct access to system's dma buffer)
4959                    <1>     ;
4960                    <1>     ;            ECX = map size in bytes
4961                    <1>     ;                (will be rounded up to page borders)
4962                    <1>     ;            EDX = Virtual Address of the buffer
4963                    <1>     ;                (Will be rounded up to page borders)
4964                    <1>     ;
4965                    <1>     ;       05/06/2017
4966                    <1>     ;       04/06/2017
4967                    <1>     ;       BH = 14 -> GET AUDIO DEVICE INFO
4968                    <1>     ;            BL: 0 = Audio Controller Info
4969                    <1>     ;              > 0 = Invalid for now!
4970                    <1>     ;
4971                    <1>     ;       22/06/2017
4972                    <1>     ;       BH = 15 -> GET CURRENT SOUND DATA (for graphics)
4973                    <1>     ;            BL: 0 -> PCM OUT data
4974                    <1>     ;              > 0 -> Invalid for now!
4975                    <1>     ;            ECX = 0 -> Get DMA Buffer Pointer
4976                    <1>     ;               EDX = Not Used
4977                    <1>     ;            ECX > 0 -> Byte count for buffer (EDX)
4978                    <1>     ;                EDX = Buffer Address (Virtual)
4979                    <1>     ;
4980                    <1>     ;       10/10/2017
4981                    <1>     ;       BH = 16 -> UPDATE DMA BUFFER DATA
4982                    <1>     ;                 (by using the Audio Buffer content)
4983                    <1>     ;            BL = 0 : Update dma half buffer in sequence
4984                    <1>     ;                     (automatic destination)
4985                    <1>     ;                1 : Update 1st half of the dma buffer
4986                    <1>     ;                2 : Update 2nd half of the dma buffer
4987                    <1>     ;                3-FEh: Invalid!
4988                    <1>     ;                  FFh = Get current flag value
4989                    <1>     ;                       (Half buffer number -1)
4990                    <1>     ;
4991                    <1>     ;
4992                    <1>     ; Outputs:
4993                    <1>     ;
4994                    <1>     ;       For BH = 0 -> Beep
4995                    <1>     ;           None
4996                    <1>     ;
4997                    <1>     ;       01/04/2017
4998                    <1>     ;
4999                    <1>     ;       For BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
5000                    <1>     ;           AH = 0 : PC SPEAKER
5001                    <1>     ;                1 : SOUND BLASTER 16
5002                    <1>     ;                2 : INTEL AC'97
5003                    <1>     ;                3 : VIA VT8237R (VT8233)
5004                    <1>     ;                4 : INTEL HDA
5005                    <1>     ;              5-FFh : unknown/invalid
5006                    <1>     ;           AL = mode status
5007                    <1>     ;                bit 0 = mono /stereo (1 = stereo)
5008                    <1>     ;                bit 1 = 8 bit / 16 bit ( 1 = 16 bit)
5009                    <1>     ;           04/06/2017
5010                    <1>     ;           EBX = PCI DEVICE/VENDOR ID (if >0)
5011                    <1>     ;               (BX = VENDOR ID)
5012                    <1>     ;           (if CF = 1 -> Error code in EAX)
5013                    <1>     ;
5014                    <1>     ;       For BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
5015                    <1>     ;           EAX = Physical Address of the buffer
5016                    <1>     ;           (if CF = 1 -> Error code in EAX)
5017                    <1>     ;
5018                    <1>     ;       For BH = 3 -> INITIALIZE AUDIO DEVICE
5019                    <1>     ;           (if CF = 1 -> Error code in EAX)
5020                    <1>     ;
5021                    <1>     ;       For BH = 4 -> START TO PLAY
5022                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5023                    <1>     ;
5024                    <1>     ;       For BH = 5 -> PAUSE
5025                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5026                    <1>     ;
5027                    <1>     ;       For BH = 6 -> CONTINUE TO PLAY
5028                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5029                    <1>     ;
5030                    <1>     ;       For BH = 7 -> STOP
5031                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5032                    <1>     ;
5033                    <1>     ;       For BH = 8 -> RESET
5034                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5035                    <1>     ;
5036                    <1>     ;       For BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
5037                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5038                    <1>     ;
5039                    <1>     ;       For BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
5040                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5041                    <1>     ;
5042                    <1>     ;       For BH = 11 -> SET VOLUME LEVEL
5043                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5044                    <1>     ;
5045                    <1>     ;       For BH = 12 -> DISABLE AUDIO DEVICE
5046                    <1>     ;           none (if CF = 1 -> Error code in EAX)
5047                    <1>     ;
5048                    <1>     ;       12/05/2017
5049                    <1>     ;       For BH = 13 -> MAP DMA BUFFER TO USER
5050                    <1>     ;           EAX = Physical Address of the buffer
5051                    <1>     ;           (if CF = 1 -> Error code in EAX)
5052                    <1>     ;
5053                    <1>     ;       04/06/2017
5054                    <1>     ;       For BH = 14 -> GET AUDIO DEVICE INFO
5055                    <1>     ;       (for BL = 0) ; 05/06/2017
5056                    <1>     ;           EAX = IRQ Number in AL
5057                    <1>     ;                Audio Device Number in AH
5058                    <1>     ;           EBX = DEV/VENDOR ID
5059                    <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>       ;      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>
0001338C 80FF11                 <1>       cmp    bh, AUDIO1L/4
0001338F 0F8337A4FFFF           <1>       jnb    sysret
                                <1>
00013395 C0E702                 <1>       shl    bh, 2 ; *4
00013398 0FB6F7                 <1>       movzx  esi, bh
                                <1>
                                <1>       ; 22/04/2017
0001339B 31C0                   <1>       xor    eax, eax
0001339D A3[64030300]           <1>       mov    [u.r0], eax  ; 0
                                <1>
000133A2 FF96[AD330100]         <1>       call   dword [esi+AUDIO1]
                                <1>       ;jc    error
000133A8 E91FA4FFFF             <1>       jmp    sysret
                                <1>
000133AD [64230000]             <1> AUDIO1:     dd     _beep ; 12/02/2021
                                <1>       ;dd    beep ; FUNCTION = 0 (bl = Duration Counter
                                <1>                   ;                    cx = Frequency Divisor
000133B1 [F1330100]             <1>       dd     soundc_detect
000133B5 [8D340100]             <1>       dd     sound_alloc
000133B9 [4B350100]             <1>       dd     soundc_init
000133BD [03370100]             <1>       dd     sound_play
000133C1 [9F370100]             <1>       dd     sound_pause
000133C5 [C9370100]             <1>       dd     sound_continue
000133C9 [F3370100]             <1>       dd     sound_stop
000133CD [1C380100]             <1>       dd     soundc_reset
000133D1 [4D380100]             <1>       dd     soundc_cancel
000133D5 [73380100]             <1>       dd     sound_dalloc
000133D9 [9E380100]             <1>       dd     sound_volume
000133DD [F0380100]             <1>       dd     soundc_disable
000133E1 [62390100]             <1>       dd     sound_dma_map
000133E5 [D1390100]             <1>       dd     soundc_info
000133E9 [303A0100]             <1>       dd     sound_data
000133ED [DD3A0100]             <1>       dd     sound_update
                                <1>
                                <1> AUDIO1L     EQU    $ - AUDIO1
                                <1>
                                <1> soundc_detect:
                                <1>       ; FUNCTION = 1
                                <1>       ; bl = Audio device type number
                                <1>       ; (0= pc speaker, 1 = sound blaster 16, 2 = intel ac97
                                <1>       ;  3= via vt823x, 4 = intel HDA, 0FFh= any)
                                <1>
                                <1>       ; 04/06/2017
000133F1 8A25[719B0100]         <1>       mov    ah, [audio_device]
000133F7 80FBFF                 <1>       cmp    bl, 0FFh ; get current audio device id
000133FA 7408                   <1>       je     short sysaudio0
                                <1>
000133FC 20E4                   <1>       and    ah, ah
000133FE 741E                   <1>       jz     short soundc_get_dev
                                <1>
00013400 38DC                   <1>       cmp    ah, bl
00013402 7567                   <1>       jne    short soundc_dev_err
                                <1>
                                <1> sysaudio0:
00013404 A0[729B0100]           <1>       mov    al, [audio_mode]
                                <1> sysaudio1:
00013409 A3[64030300]           <1>       mov    [u.r0], eax
0001340E 8B1D[7C9B0100]         <1>       mov    ebx, [audio_vendor] ; (DEVICE/VENDOR ID)
00013414 8B2D[60030300]         <1>       mov    ebp, [u.usp]
0001341A 895D10                 <1>       mov    [ebp+16], ebx  ; ebx
0001341D C3                     <1>       retn
                                <1>
                                <1> soundc_get_dev:
                                <1>       ; 28/05/2017
                                <1>       ; 03/04/2017, 24/04/2017
0001341E C605[709B0100]00       <1>       mov    byte [audio_pci], 0
00013425 80FB03                 <1>       cmp    bl, 3 ; VIA VT8233 (VT8237R) Audio Controller & AC97 Codec
                                <1>       ;jne   short soundc_get_dev_sb
                                <1>       ; 28/05/2017
00013428 7220                   <1>       jb     short soundc_get_dev_sb
0001342A 773F                   <1>       ja     short soundc_dev_err  ; temporary (28/05/2017)
                                <1>       ;
0001342C E86C180000             <1>       call   DetectVT8233
00013431 7238                   <1>       jc     short soundc_dev_err
```

```
                                <1>       ; eax = 0
                                <1>
                                <1>       ;mov   ebx, [audio_vendor]
                                <1>       ; ebx = DEVICE/VENDOR ID
                                <1>       ;      DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
                                <1>
00013433 B003                   <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
00013435 88C4                   <1>       mov    ah, al
                                <1>
                                <1> soundc_get_pci_dev_ok: ; 28/05/2017
00013437 FE05[709B0100]         <1>       inc    byte [audio_pci] ; = 1
                                <1> soundc_get_dev_ok:
                                <1>
                                <1> soundc_get_dev_sb16_ok:
0001343D A2[719B0100]           <1>       mov    [audio_device], al
00013442 8825[729B0100]         <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
00013448 EBBF                   <1>       jmp    short sysaudio1
                                <1>
                                <1> soundc_get_dev_sb:
                                <1>       ; 24/04/2017
0001344A 80FB01                 <1>       cmp    bl, 1 ; Sound Blaster 16
0001344D 750E                   <1>       jne    short soundc_get_dev_ich ; 28/05/2017
                                <1>       ;
0001344F E86D1D0000             <1>       call   DetectSB
00013454 7215                   <1>       jc     short soundc_dev_err
00013456 B801030000             <1>       mov    eax, 0301h ; Sound Blaster 16
0001345B EBE0                   <1>       jmp    short soundc_get_dev_sb16_ok
                                <1>
                                <1> soundc_get_dev_ich:
                                <1>       ; 28/05/2017
                                <1>       ;cmp   bl, 2 ; Intel AC'97 Audio Controller (ICH)
                                <1>       ;jne   short soundc_dev_err ; Temporary  (28/05/2017)
                                <1>       ;                           ; (Here will be modified just after
                                <1>       ;                           ; new sound card code will be ready!)
0001345D E82E180000             <1>       call   DetectICH
00013462 7207                   <1>       jc     short soundc_dev_err
                                <1>       ;
00013464 B802030000             <1>       mov    eax, 0302h ; AC'97 (ICH)
00013469 EBCC                   <1>       jmp    short soundc_get_pci_dev_ok
                                <1>
                                <1> soundc_dev_err:
0001346B B80F000000             <1>       mov    eax, ERR_DEV_NOT_RDY ; Device not ready !
00013470 EB0C                   <1>       jmp    short sysaudio_err
                                <1>
                                <1> sound_buff_error:
00013472 B82E000000             <1>       mov    eax, ERR_BUFFER ; Buffer error !
00013477 EB05                   <1>       jmp    short sysaudio_err
                                <1>
                                <1> soundc_respond_err:
                                <1>       ; ERR_TIME_OUT ; 'time out !' error
00013479 B819000000             <1>       mov    eax, ERR_DEV_NOT_RESP ; 'device not responding !' error
                                <1> sysaudio_err:
0001347E A3[64030300]           <1>       mov    [u.r0], eax
00013483 A3[C8030300]           <1>       mov    [u.error], eax
00013488 E91FA3FFFF             <1>       jmp    error
                                <1>
                                <1> sound_alloc:
                                <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
0001348D 803D[709B0100]00       <1>       cmp    byte [audio_pci], 0
00013494 7708                   <1>       ja     short snd_alloc_0
                                <1>       ; Max. 64KB DMA buffer !!!
00013496 81F900800000           <1>       cmp    ecx, 32768
0001349C 77D4                   <1>       ja     short sound_buff_error
                                <1> snd_alloc_0:
                                <1>       ; 15/05/2017
0001349E 81F900100000           <1>       cmp    ecx, 4096 ; PAGE_SIZE
000134A4 72CC                   <1>       jb     short sound_buff_error
                                <1>       ;
000134A6 A1[849B0100]           <1>       mov    eax, [audio_buffer] ; audio buffer address (current)
000134AB 09C0                   <1>       or     eax, eax
000134AD 7445                   <1>       jz     short snd_alloc_2
                                <1>       ; audio buffer exists !
000134AF 8A1D[B3030300]         <1>       mov    bl, [u.uno]
000134B5 3A1D[999B0100]         <1>       cmp    bl, [audio_user]
000134BB 0F85FC000000           <1>       jne    sndc_owner_error ; not owner !
000134C1 39D0                   <1>       cmp    eax, edx ; same virtual buffer address ?
000134C3 7508                   <1>       jne    short snd_alloc_1
000134C5 3B0D[8C9B0100]         <1>       cmp    ecx, [audio_buff_size]
000134CB 746C                   <1>       je     short snd_alloc_3 ; Nothing to do !
                                <1>                                ; Buffer has been set already!
                                <1> snd_alloc_1:
000134CD 51                     <1>       push   ecx
000134CE 52                     <1>       push   edx
000134CF 89C3                   <1>       mov    ebx, eax ; audio buffer address (current)
000134D1 8B0D[8C9B0100]         <1>       mov    ecx, [audio_buff_size]
000134D7 E8D031FFFF             <1>       call   deallocate_user_pages
000134DC 5A                     <1>       pop    edx
000134DD 59                     <1>       pop    ecx
000134DE 31C0                   <1>       xor    eax, eax ; 0
000134E0 A3[849B0100]           <1>       mov    [audio_buffer], eax  ; 0
000134E5 A3[889B0100]           <1>       mov    [audio_p_buffer], eax  ; 0
000134EA A3[8C9B0100]           <1>       mov    [audio_buff_size], eax
000134EF A2[999B0100]           <1>       mov    [audio_user], al ; 0
                                <1> snd_alloc_2:
000134F4 89D3                   <1>       mov    ebx, edx
                                <1>       ; 01/05/2017
000134F6 BA00F0FFFF             <1>       mov    edx, ~PAGE_OFF ; truncating page offsets
                                <1>                             ; for aligning to page borders
                                <1>       ;and   eax, edx
```

```
5270 000134FB 21D3               <1>       and    ebx, edx
5271 000134FD 21D1               <1>       and    ecx, edx
5272                             <1>       ; 15/05/2017
5273                             <1>       ; EAX = Beginning address (physical)
5274                             <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
5275                             <1>       ; ECX = Number of bytes to be allocated
5276 000134FF E84D2EFFFF         <1>       call   allocate_memory_block
5277 00013504 0F8268FFFFFF       <1>       jc     sound_buff_error
5278                             <1>       ; EAX = Physical address of the allocated memory block
5279                             <1>       ; ECX = Allocated bytes (as truncated to page border)
5280                             <1>       ; EBX = Virtual address (as truncated to page border)
5281 0001350A 50                 <1>       push   eax
5282 0001350B 53                 <1>       push   ebx
5283 0001350C 51                 <1>       push   ecx
5284 0001350D E88F32FFFF         <1>       call   allocate_user_pages
5285 00013512 59                 <1>       pop    ecx
5286 00013513 5B                 <1>       pop    ebx
5287 00013514 58                 <1>       pop    eax
5288 00013515 722A               <1>       jc     short snd_alloc_4  ; insufficient memory, buff error
5289                             <1>       ; eax = physical address of the user's audio buffer
5290                             <1>       ; ebx = virtual address of the user's audio buffer
5291                             <1>       ; ecx = buffer size (in bytes)
5292 00013517 A3[889B0100]       <1>       mov    [audio_p_buffer], eax
5293 0001351C 891D[849B0100]     <1>       mov    [audio_buffer], ebx
5294 00013522 890D[8C9B0100]     <1>       mov    [audio_buff_size], ecx
5295 00013528 8A15[B3030300]     <1>       mov    dl, [u.uno]
5296 0001352E 8815[999B0100]     <1>       mov    [audio_user], dl
5297 00013534 A3[64030300]       <1>       mov    [u.r0], eax
5298                             <1> snd_alloc_3:
5299                             <1>       ; 27/07/2020
5300 00013539 C605[989B0100]00   <1>       mov    byte [audio_flag], 0 ; clear dma half buffer flag
5301                             <1>       ;
5302 00013540 C3                 <1>       retn
5303                             <1> snd_alloc_4:
5304                             <1>       ; 15/05/2017
5305                             <1>       ; EAX = Beginning address (physical)
5306                             <1>       ; ECX = Number of bytes to be deallocated
5307 00013541 E81830FFFF         <1>       call   deallocate_memory_block
5308 00013546 E927FFFFFF         <1>       jmp    sound_buff_error  ; insufficient memory, buff error
5309                             <1>
5310                             <1> soundc_init:
5311                             <1>       ; FUNCTION = 3
5312                             <1>       ; bl = method (0= s.r.b., 1= callback, 2= auto incr s.r.b.)
5313                             <1>       ; cl = signal response byte (initial or fixed) value
5314                             <1>       ; edx = signal response byte or callback address
5315                             <1>       ; 27/07/2020
5316                             <1>       ; 28/05/2017
5317                             <1>       ; 12/05/2017, 20/05/2017
5318                             <1>       ; 22/04/2017, 23/04/2017, 24/04/2017
5319                             <1>       ; 13/04/2017, 14/04/2017, 16/04/2017, 21/04/2017
5320                             <1>       ; 03/04/2017, 10/04/2017
5321                             <1>
5322 0001354B A0[719B0100]       <1>       mov    al, [audio_device]
5323 00013550 20C0               <1>       and    al, al
5324 00013552 7549               <1>       jnz    short sndc_init_6
5325                             <1>       ;
5326 00013554 C605[709B0100]00   <1>       mov    byte [audio_pci], 0
5327 0001355B 52                 <1>       push   edx
5328 0001355C 53                 <1>       push   ebx
5329 0001355D 51                 <1>       push   ecx
5330 0001355E E85E1C0000         <1>       call   DetectSB
5331 00013563 7213               <1>       jc     short sndc_init_8
5332 00013565 66B80103           <1>       mov    ax, 0301h ; Sound Blaster 16
5333 00013569 EB1E               <1>       jmp    short sndc_init_7
5334                             <1>
5335                             <1> sndc_init_11:
5336                             <1>       ; 28/05/2017
5337 0001356B E820170000         <1>       call   DetectICH ; Detect AC'97 (ICH) Audio Controller
5338 00013570 7217               <1>       jc     short sndc_init_7
5339 00013572 66B80203           <1>       mov    ax, 0302h ; Intel AC'97 Audio Device
5340 00013576 EB0B               <1>       jmp    short sndc_init_12 ; (PCI device)
5341                             <1>
5342                             <1> sndc_init_8:
5343 00013578 E820170000         <1>       call   DetectVT8233
5344                             <1>       ;jc    short sndc_init_7
5345 0001357D 72EC               <1>       jc     sndc_init_11 ; 28/05/2017
5346                             <1>       ; eax = 0
5347 0001357F B003               <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
5348 00013581 88C4               <1>       mov    ah, al
5349                             <1>
5350                             <1> sndc_init_12:
5351 00013583 FE05[709B0100]     <1>       inc    byte [audio_pci] ; = 1
5352                             <1> sndc_init_7:
5353 00013589 59                 <1>       pop    ecx
5354 0001358A 5B                 <1>       pop    ebx
5355 0001358B 5A                 <1>       pop    edx
5356 0001358C 0F82D9FEFFFF       <1>       jc     soundc_dev_err
5357                             <1>       ;
5358 00013592 A2[719B0100]       <1>       mov    [audio_device], al
5359 00013597 8825[729B0100]     <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
5360                             <1>
5361                             <1> sndc_init_6:
5362 0001359D 833D[849B0100]00   <1>       cmp    dword [audio_buffer], 0
5363 000135A4 0F86C8FEFFFF       <1>       jna    sound_buff_error
5364                             <1>
5365 000135AA A0[B3030300]       <1>       mov    al, [u.uno]
5366 000135AF 8A25[999B0100]     <1>       mov    ah, [audio_user]
5367 000135B5 08E4               <1>       or     ah, ah
5368 000135B7 7418               <1>       jz     short sndc_init0
5369 000135B9 38E0               <1>       cmp    al, ah
5370 000135BB 7419               <1>       je     short sndc_init1
5371                             <1>
5372                             <1> sndc_owner_error:
5373 000135BD B80B000000         <1>       mov    eax, ERR_NOT_OWNER ; 'permission denied !' error
5374                             <1> sndc_perm_error:
```

```
5375 000135C2 A3[64030300]        <1>       mov   [u.r0], eax
5376 000135C7 A3[C8030300]        <1>       mov   [u.error], eax
5377 000135CC E9DBA1FFFF          <1>       jmp   error
5378                              <1> sndc_init0:
5379 000135D1 A2[999B0100]        <1>       mov   [audio_user], al
5380                              <1> sndc_init1:
5381 000135D6 8915[9C9B0100]      <1>       mov   [audio_cb_addr], edx
5382 000135DC 881D[9A9B0100]      <1>       mov   [audio_cb_mode], bl
5383 000135E2 880D[9B9B0100]      <1>       mov   [audio_srb], cl
5384                              <1>
5385                              <1>       ; 27/07/2020
5386                              <1>       ;mov  byte [audio_flag], 0  ; clear dma half buffer flag
5387                              <1>
5388                              <1>       ; 24/04/2017
5389 000135E8 803D[719B0100]03    <1>       cmp   byte [audio_device], 3 ; VT8233 (VT8237R)
5390 000135EF 7438                <1>       je    short sndc_init_9
5391                              <1>       ;ja   short soundc_respond_err ; temporary (28/05/2017)
5392 000135F1 803D[719B0100]01    <1>       cmp   byte [audio_device], 1 ; SB 16
5393 000135F8 7510                <1>       jne   short sndc_init_13
5394 000135FA BB[E6530100]        <1>       mov   ebx, sb16_int_handler
5395                              <1>       ; Note: 'SbInit' is at 'Start to Play' stage
5396                              <1>       ; 20/05/2017
5397 000135FF 66C705[A69B0100]08- <1>       mov   word [audio_master_volume], 0808h ; 2/8
5397 00013607 08                  <1>
5398 00013608 EB3F                <1>       jmp   short sndc_init_10
5399                              <1> sndc_init_13:
5400                              <1>       ; 28/05/2017
5401 0001360A 803D[719B0100]02    <1>       cmp   byte [audio_device], 2 ; AC 97 (ICH)
5402 00013611 0F8562FEFFFF        <1>       jne   soundc_respond_err ; temporary (28/05/2017)
5403                              <1>
5404 00013617 E81F1F0000          <1>       call  ac97_codec_config
5405 0001361C 0F8257FEFFFF        <1>       jc    soundc_respond_err ; codec error !
5406                              <1>
5407 00013622 BB[22570100]        <1>       mov   ebx, ac97_int_handler
5408 00013627 EB20                <1>       jmp   short sndc_init_10
5409                              <1>
5410                              <1> sndc_init_9:
5411                              <1>       ;call  reset_codec
5412                              <1>       ;; eax = 1
5413                              <1>       ;call  codec_io_w16 ; w32
5414 00013629 E8E6170000          <1>       call  init_codec ; 28/05/2017
5415 0001362E 0F8245FEFFFF        <1>       jc    soundc_respond_err ; codec error !
5416                              <1>
5417 00013634 E80D1A0000          <1>       call  channel_reset
5418                              <1>
5419                              <1>       ; setup the Codec (actually mixer registers)
5420 00013639 E82D190000          <1>         call    codec_config  ; unmute codec, set rates.
5421 0001363E 0F8235FEFFFF        <1>       jc    soundc_respond_err ; codec error !
5422                              <1>
5423 00013644 BB[D84F0100]        <1>       mov   ebx, vt8233_int_handler
5424                              <1> sndc_init_10:
5425                              <1>       ; 13/04/2017
5426 00013649 A0[739B0100]        <1>       mov   al, [audio_intr] ; IRQ number
5427 0001364E E82BFDFFFF          <1>       call  set_dev_IRQ_service
5428                              <1>
5429                              <1>       ; SETUP (audio) INTERRUPT CALLBACK SERVICE
5430 00013653 8A1D[739B0100]      <1>       mov   bl, [audio_intr] ; IRQ number
5431 00013659 8A3D[9A9B0100]      <1>       mov   bh, [audio_cb_mode]
5432 0001365F FEC7                <1>       inc   bh  ; 1 = Signal Response Byte method (fixed value)
5433                              <1>                 ; 2 = Callback service method
5434                              <1>                 ; 3 = Auto Increment S.R.B. method
5435 00013661 8A0D[9B9B0100]      <1>       mov   cl, [audio_srb]
5436 00013667 8B15[9C9B0100]      <1>       mov   edx, [audio_cb_addr]
5437 0001366D A0[999B0100]        <1>       mov   al, [audio_user]
5438                              <1>       ; 14/04/2017
5439 00013672 E8E1040000          <1>       call  set_irq_callback_service
5440                              <1>       ; 16/04/2017
5441 00013677 A3[64030300]        <1>       mov   [u.r0], eax
5442                              <1>       ;jnc  sysret
5443 0001367C 7316                <1>       jnc   short sndc_init2 ; 21/04/2017
5444                              <1>       ;
5445 0001367E A3[C8030300]        <1>       mov   dword [u.error], eax
5446                              <1>
5447 00013683 A0[739B0100]        <1>       mov   al, [audio_intr] ; IRQ number
5448 00013688 31DB                <1>       xor   ebx, ebx ; reset IRQ handler address
5449 0001368A E8EFFCFFFF          <1>       call  set_dev_IRQ_service
5450                              <1>
5451 0001368F E918A1FFFF          <1>       jmp   error
5452                              <1>
5453                              <1> sndc_init2:
5454                              <1>       ; 21/04/2017
5455 00013694 8B0D[8C9B0100]      <1>       mov   ecx, [audio_buff_size] ; audio buffer size
5456 0001369A D1E1                <1>       shl   ecx, 1 ; *2
5457 0001369C A1[909B0100]        <1>       mov   eax, [audio_dma_buff]
5458 000136A1 21C0                <1>       and   eax, eax
5459 000136A3 7415                <1>       jz    short sndc_init3
5460                              <1>
5461 000136A5 8B15[949B0100]      <1>       mov   edx, [audio_dmabuff_size] ; dma buffer size
5462 000136AB 39D1                <1>       cmp   ecx, edx
5463 000136AD 744D                <1>       je    short sndc_init5
5464                              <1>
5465 000136AF 87CA                <1>       xchg  ecx, edx
5466 000136B1 E8A82EFFFF          <1>       call  deallocate_memory_block
5467 000136B6 87D1                <1>       xchg  edx, ecx
5468 000136B8 31C0                <1>       xor   eax, eax
5469                              <1> sndc_init3:
5470                              <1>       ; 12/05/2017
5471 000136BA 803D[719B0100]01    <1>       cmp   byte [audio_device], 1 ; SB 16
5472 000136C1 7515                <1>       jne   short sndc_init4
5473 000136C3 C705[909B0100]-     <1>       mov   dword [audio_dma_buff], sb16_dma_buffer
5473 000136C9 [00000200]          <1>
5474 000136CD C705[949B0100]0000- <1>       mov   dword [audio_dmabuff_size], 65536
5474 000136D5 0100                <1>
5475                              <1>       ;xor  eax, eax
5476                              <1>       ;mov  [u.r0], eax ; 0 = no error, successful
```

```
5477 000136D7 C3                <1>       retn
5478                            <1>
5479                            <1> sndc_init4:
5480                            <1>       ; EAX = Beginning address (physical)
5481                            <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
5482                            <1>       ; ECX = Number of bytes to be allocated (>0)
5483 000136D8 E8742CFFFF        <1>       call   allocate_memory_block
5484 000136DD 0F828FFDFFFF      <1>       jc     sound_buff_error
5485                            <1>
5486                            <1>       ; set dma buffer address and size parameters
5487 000136E3 A3[909B0100]      <1>       mov    [audio_dma_buff], eax ; dma buffer address
5488 000136E8 890D[949B0100]    <1>       mov    [audio_dmabuff_size], ecx ; dma buffer size
5489                            <1> ;     ; EAX = Beginning (physical) addr of the allocated mem block
5490                            <1> ;     ; ECX = Num of allocated bytes (rounded up to page borders)
5491                            <1> ;     cmp    byte [audio_pci], 0 ; AC97 audio controller ?
5492                            <1> ;     ja     short sndc_init4
5493                            <1> ;
5494                            <1> ;     ; Sound Blaster 16 uses classic DMA
5495                            <1> ;     mov    edx, eax
5496                            <1> ;     add    edx, ecx
5497                            <1> ;     cmp    edx, 1000000h ; 1st 16 MB
5498                            <1> ;     jna    short sndc_init4
5499                            <1> ;
5500                            <1> ;     ; error !
5501                            <1> ;     ; restore Memory Allocation Table Content
5502                            <1> ;     ; EAX = Beginning address (physical)
5503                            <1> ;     ; ECX = Number of bytes to be deallocated
5504                            <1> ;     call   deallocate_memory_block
5505                            <1> ;     ; reset dma buffer address and size parameters
5506                            <1> ;     xor    eax, eax ; 0
5507                            <1> ;     mov    [audio_dma_buff], eax ; 0
5508                            <1> ;     mov    [audio_dmabuff_size], ecx ; 0
5509                            <1> ;     jmp    sound_buff_error
5510                            <1> ;
5511                            <1> ;sndc_init4:
5512 000136EE 803D[719B0100]03  <1>       cmp    byte [audio_device], 3
5513                            <1>       ;jne   short sndc_init5
5514 000136F5 7506              <1>       jne    short sndc_init14 ; 28/05/2017
5515 000136F7 E88B190000        <1>       call   set_vt8233_bdl
5516                            <1> sndc_init5:
5517                            <1>       ;sub   eax, eax ; 0
5518                            <1>       ;mov   [u.r0], eax ; 0 = no error, successful
5519 000136FC C3                <1>       retn
5520                            <1> sndc_init14:
5521 000136FD E8521F0000        <1>       call   set_ac97_bdl
5522                            <1>       ;jmp   short sndc_init5
5523 00013702 C3                <1>       retn
5524                            <1>
5525                            <1> sound_play:
5526                            <1>       ; FUNCTION = 4
5527                            <1>       ; bl = Mode
5528                            <1>       ;      bit 0 = mono/stereo (1 = stereo)
5529                            <1>       ;      bit 1 = 8 bit / 16 bit (1 = 16 bit)
5530                            <1>       ; cx = Sampling Rate (Hz)
5531                            <1>
5532                            <1>       ; 13/06/2017
5533                            <1>       ; Note: Even if Mode bits are not 11b,
5534                            <1>       ;     AC'97 Audio Controller (&Codec)
5535                            <1>       ;     will play audio samples as 16 bit, stereo
5536                            <1>       ;     samples.
5537                            <1>       ;     (Program must fill the audio buffer
5538                            <1>       ;     as required; 8 bit samples must be converted
5539                            <1>       ;     to 16 bit samples and mono samples must be
5540                            <1>       ;     converted to stereo samples...)
5541                            <1>       ;
5542                            <1>       ; 28/07/2020
5543                            <1>       ; 27/07/2020
5544                            <1>       ; 28/05/2017
5545                            <1>       ; 15/05/2017, 20/05/2017
5546                            <1>       ; 21/04/2017, 24/04/2017
5547                            <1>       ; ... device check at first
5548 00013703 A0[719B0100]      <1>       mov    al, [audio_device]
5549 00013708 08C0              <1>       or     al, al ; 0 ; pc speaker or invalid
5550 0001370A 0F8465ECFEFF      <1>       jz     beeper_gfx ; 'video.s' ; temporary !
5551                            <1> ;     cmp    al, 3 ; VIA VT 8237R (vt8233)
5552                            <1> ;     je     short snd_play_1
5553                            <1> ;     cmp    al, 1 ; SB 16
5554                            <1> ;     jne    soundc_dev_err ; temporary !
5555                            <1> ;snd_play_0:
5556                            <1>       ; ... buffer & (buffer) owner check at second
5557 00013710 833D[849B0100]00  <1>       cmp    dword [audio_buffer], 0
5558 00013717 0F8655FDFFFF      <1>       jna    sound_buff_error
5559 0001371D A0[B3030300]      <1>       mov    al, [u.uno]
5560 00013722 3A05[999B0100]    <1>       cmp    al, [audio_user]
5561 00013728 0F858FFEFFFF      <1>       jne    sndc_owner_error
5562                            <1>
5563 0001372E 66890D[A29B0100]  <1>       mov    [audio_freq], cx ; sample frequency (Hertz)
5564 00013735 88D8              <1>       mov    al, bl
5565 00013737 2401              <1>       and    al, 1 ; mono/stereo (1= stereo)
5566 00013739 FEC0              <1>       inc    al ; channels
5567 0001373B A2[A19B0100]      <1>       mov    [audio_stmo], al ; sound channels (1 or 2)
5568 00013740 B008              <1>       mov    al, 8
5569 00013742 F6C302            <1>       test   bl, 2 ; bits per sample (1= 16 bit)
5570 00013745 7402              <1>       jz     short snd_play_bps
5571 00013747 D0E0              <1>       shl    al, 1
5572                            <1> snd_play_bps:
5573 00013749 A2[A09B0100]      <1>       mov    [audio_bps], al
5574                            <1>
5575                            <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
5576 0001374E 8B3D[909B0100]    <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
5577 00013754 09FF              <1>       or     edi, edi
5578 00013756 0F8416FDFFFF      <1>       jz     sound_buff_error
5579                            <1>
5580                            <1>       ; 27/07/2020
5581                            <1>
```

```
5582 0001375C 8B35[889B0100]      <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
5583                              <1>       ;mov   ecx, [audio_buff_size] ; 15/05/2017
5584 00013762 8B0D[949B0100]      <1>       mov    ecx, [audio_dmabuff_size] ; 27/07/2020
5585                              <1>       ;or    ecx, ecx
5586                              <1>       ;jz    sound_buff_error
5587                              <1>       ; 28/07/2020
5588 00013768 D1E9                <1>       shr    ecx, 1  ; dma half buffer size
5589                              <1>
5590 0001376A 8035[989B0100]01    <1>       xor    byte [audio_flag], 1 ;  0 -> 1, 1 -> 0
5591 00013771 7502                <1>       jnz    short snd_play_0 ; [audio_flag] = 1
5592                              <1>                               ; fill dma half buffer 1
5593                              <1>       ; [audio_flag] = 0
5594                              <1>
5595                              <1>       ; fill dma half buffer 2
5596 00013773 01CF                <1>       add    edi, ecx
5597                              <1>
5598                              <1> snd_play_0:
5599                              <1>       ;rep   movsb
5600 00013775 C1E902              <1>       shr    ecx, 2  ; convert byte count to dword count
5601 00013778 F3A5                <1>       rep    movsd
5602                              <1>
5603                              <1>       ; here, if [audio_flag] = 0, interrupt handler will update
5604                              <1>                           ; dma half buffer 2
5605                              <1>                           ; (user's audio buffer data will be
5606                              <1>                           ; copied into dma half buffer 2)
5607                              <1>       ;; 20/05/2017
5608                              <1>       ;mov   byte [audio_flag], 1 ; next half (on next time)
5609                              <1>
5610                              <1>       ; 24/04/2017
5611 0001377A A0[719B0100]        <1>       mov    al, [audio_device]
5612 0001377F 3C03                <1>       cmp    al, 3 ; VT8233 (VT8237R)
5613 00013781 7410                <1>       je     short snd_play_1
5614 00013783 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
5615 00013785 7512                <1>       jne    short snd_play_2  ; 28/05/2017
5616 00013787 E8031B0000          <1>       call   SbInit_play
5617 0001378C 0F82E7FCFFFF        <1>       jc     soundc_respond_err
5618 00013792 C3                  <1>       retn
5619                              <1>
5620                              <1> snd_play_1:
5621 00013793 E826190000          <1>       call   vt8233_start_play
5622 00013798 C3                  <1>       retn
5623                              <1>
5624                              <1> snd_play_2:
5625                              <1>       ; 28/05/2017
5626                              <1>       ;cmp   al, 2 ; AC'97
5627                              <1>       ;jne   short snd_play_3
5628                              <1>
5629 00013799 E8EA1E0000          <1>       call   ac97_start_play
5630 0001379E C3                  <1>       retn
5631                              <1>
5632                              <1> ;snd_play_3:
5633                              <1> ;     ;call  hda_start_play
5634                              <1> ;     retn
5635                              <1>
5636                              <1> sound_pause:
5637                              <1>       ; FUNCTION = 5
5638                              <1>       ; Pause
5639                              <1>       ; 28/05/2017
5640                              <1>       ; 24/04/2017
5641                              <1>       ; 22/04/2017
5642 0001379F E814030000          <1>       call   snd_dev_check
5643 000137A4 7275                <1>       jc     short snd_nothing ; temporary.
5644 000137A6 E81A030000          <1>       call   snd_buf_check
5645 000137AB 726E                <1>       jc     short snd_nothing ; temporary.
5646 000137AD A0[719B0100]        <1>       mov    al, [audio_device]
5647 000137B2 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
5648 000137B4 7409                <1>       je     short snd_pause_1
5649 000137B6 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
5650 000137B8 750A                <1>       jne    short snd_pause_2 ; 28/05/2017
5651 000137BA E9AE1C0000          <1>       jmp    sb16_pause
5652                              <1> snd_pause_1:
5653 000137BF E9B3190000          <1>       jmp    vt8233_pause
5654                              <1> snd_pause_2:
5655                              <1>       ; 28/05/2017
5656                              <1>       ;cmp   al, 2 ; AC'97
5657                              <1>       ;jne   short snd_nothing ; temporary.
5658 000137C4 E94D200000          <1>       jmp    ac97_pause
5659                              <1>
5660                              <1> sound_continue:
5661                              <1>       ; FUNCTION = 6
5662                              <1>       ; Continue to play
5663                              <1>       ; 28/05/2017
5664                              <1>       ; 22/04/2017
5665 000137C9 E8EA020000          <1>       call   snd_dev_check
5666 000137CE 724B                <1>       jc     short snd_nothing ; temporary.
5667 000137D0 E8F0020000          <1>       call   snd_buf_check
5668 000137D5 7244                <1>       jc     short snd_nothing ; temporary.
5669 000137D7 A0[719B0100]        <1>       mov    al, [audio_device]
5670 000137DC 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
5671 000137DE 7409                <1>       je     short snd_cont_1
5672 000137E0 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
5673 000137E2 750A                <1>       jne    short snd_cont_2 ; 28/05/2017
5674 000137E4 E9A71C0000          <1>       jmp    sb16_continue
5675                              <1> snd_cont_1:
5676 000137E9 E93A190000          <1>       jmp    vt8233_play
5677                              <1> snd_cont_2:
5678                              <1>       ; 28/05/2017
5679                              <1>       ;cmp   al, 2 ; AC'97
5680                              <1>       ;jne   short snd_nothing ; temporary.
5681 000137EE E9EB1E0000          <1>       jmp    ac97_play
5682                              <1>
5683                              <1> sound_stop:
5684                              <1>       ; FUNCTION = 7
5685                              <1>       ; Stop playing
5686                              <1>       ; 28/05/2017
```

```
                                <1>       ; 24/05/2017
                                <1>       ; 21/04/2017, 22/04/2017, 24/04/2017
000137F3 E8C0020000             <1>       call   snd_dev_check
000137F8 7221                   <1>       jc     short snd_nothing ; temporary.
                                <1>       ;call  snd_buf_check
000137FA E8CF020000             <1>       call   snd_user_check ; 24/05/2017
000137FF 721A                   <1>       jc     short snd_nothing ; temporary.
                                <1>
00013801 A0[719B0100]           <1>       mov    al, [audio_device]
00013806 3C03                   <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
00013808 0F8470180000           <1>       je     vt8233_stop
                                <1>       ; 28/05/2017
                                <1>       ;ja    short snd_nothing
0001380E 3C01                   <1>       cmp    al, 1 ; Sound Blaster 16
00013810 0F849D1C0000           <1>       je     sb16_stop
                                <1>       ;cmp   al, 2
                                <1>       ;je    short ac97_stop
00013816 E9CD1F0000             <1>       jmp    ac97_stop ; temporary.
                                <1>       ;jmp   hda_stop
                                <1>
                                <1> snd_nothing:
                                <1>       ; 21/04/2017
0001381B C3                     <1>       retn
                                <1>
                                <1> soundc_reset:
                                <1>       ; FUNCTION = 8
                                <1>       ; Reset Audio Controller
                                <1>       ; 28/05/2017
                                <1>       ; 22/04/2017
0001381C E897020000             <1>       call   snd_dev_check
00013821 72F8                   <1>       jc     snd_nothing ; temporary.
00013823 E89D020000             <1>       call   snd_buf_check
00013828 72F1                   <1>       jc     snd_nothing ; temporary.
                                <1>
0001382A A0[719B0100]           <1>       mov    al, [audio_device]
0001382F 3C03                   <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
00013831 0F844C190000           <1>       je     vt8233_reset
00013837 77E2                   <1>       ja     short snd_nothing ; temporary.
                                <1>       ;ja    hda_reset
00013839 3C01                   <1>       cmp    al, 1 ; Sound Blaster 16
0001383B 0F8526200000           <1>       jne    ac97_reset
00013841 E8BF1C0000             <1>       call   sb16_reset
00013846 0F822DFCFFFF           <1>       jc     soundc_respond_err
0001384C C3                     <1>       retn
                                <1>
                                <1> soundc_cancel:
                                <1>       ; FUNCTION = 9
                                <1>       ; Cancel audio callback service
                                <1>       ; 22/04/2017
0001384D A0[999B0100]           <1>       mov    al, [audio_user]
00013852 3A05[B3030300]         <1>       cmp    al, [u.uno]
00013858 75C1                   <1>       jne    short snd_nothing
                                <1>       ; RESET (audio) INTERRUPT CALLBACK SERVICE
0001385A 8A1D[739B0100]         <1>       mov    bl, [audio_intr] ; IRQ number
00013860 A0[B3030300]           <1>       mov    al, [u.uno]
00013865 28FF                   <1>       sub    bh, bh ; 0 ; unlink IRQ from user service
00013867 E8EC020000             <1>       call   set_irq_callback_service
0001386C 0F8250FDFFFF           <1>       jc     sndc_perm_error ; 'permission denied' error
00013872 C3                     <1>       retn
                                <1>
                                <1> sound_dalloc:
                                <1>       ; FUNCTION = 10
                                <1>       ; Deallocate (ring 3) audio buffer
                                <1>       ; 22/04/2017
00013873 A0[999B0100]           <1>       mov    al, [audio_user]
00013878 3A05[B3030300]         <1>       cmp    al, [u.uno]
0001387E 759B                   <1>       jne    short snd_nothing
00013880 8B1D[849B0100]         <1>       mov    ebx, [audio_buffer]
                                <1>       ;or    ebx, ebx
                                <1>       ;jz    short snd_nothing
00013886 8B0D[8C9B0100]         <1>       mov    ecx, [audio_buff_size]
0001388C E81B2EFFFF             <1>       call   deallocate_user_pages
00013891 31C0                   <1>       xor    eax, eax
00013893 A3[849B0100]           <1>       mov    [audio_buffer], eax ; 0
00013898 A2[999B0100]           <1>       mov    [audio_user], al ; 0
0001389D C3                     <1>       retn
                                <1>
                                <1> sound_volume:
                                <1>       ; FUNCTION = 11
                                <1>       ; Set sound volume level
                                <1>       ; 28/05/2017
                                <1>       ; 20/05/2017
                                <1>       ; 22/04/2017, 24/04/2017
                                <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>
0001389E 80FB80                 <1>       cmp    bl, 80h
000138A1 720E                   <1>       jb     short snd_vol_1
000138A3 0F8772FFFFFF           <1>       ja     snd_nothing ; temporary.
                                <1>       ; Set volume level for next play (BL>= 80h)
000138A9 66890D[A69B0100]       <1>       mov    [audio_master_volume], cx
000138B0 C3                     <1>       retn
                                <1> snd_vol_1:
                                <1>       ; set volume level immediate (BL< 80h)
000138B1 80FB00                 <1>       cmp    bl, 0
000138B4 0F8761FFFFFF           <1>       ja     snd_nothing ; temporary.
                                <1>
000138BA E8F9010000             <1>       call   snd_dev_check
000138BF 0F8256FFFFFF           <1>       jc     snd_nothing ; temporary.
000138C5 E8FB010000             <1>       call   snd_buf_check
000138CA 0F824BFFFFFF           <1>       jc     snd_nothing ; temporary.
                                <1>
000138D0 A0[719B0100]           <1>       mov    al, [audio_device]
000138D5 3C03                   <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
```

```
5792 000138D7 0F84BF180000        <1>       je    vt8233_volume
5793                              <1>       ; 28/05/2017
5794 000138DD 0F8738FFFFFF        <1>       ja    snd_nothing ; temporary.
5795                              <1>       ;ja   hda_volume
5796                              <1>       ; Sound Blaster 16
5797 000138E3 3C01                <1>       cmp   al, 1 ; SB 16
5798 000138E5 0F844D1B0000        <1>       je    sb16_volume
5799 000138EB E90A1E0000          <1>       jmp   ac97_volume
5800                              <1>
5801                              <1> soundc_disable:
5802                              <1>       ; FUNCTION = 12
5803                              <1>       ; Disable audio device (and unlink DMA memory)
5804                              <1>       ; 28/05/2017
5805                              <1>       ; 24/05/2017
5806                              <1>       ; 22/04/2017
5807 000138F0 E8C3010000          <1>       call  snd_dev_check
5808 000138F5 0F8270FBFFFF        <1>       jc    soundc_dev_err ; temporary.
5809                              <1>       ;call snd_buf_check
5810                              <1>       ;jc   sndc_owner_error ; temporary.
5811                              <1>
5812 000138FB A0[719B0100]        <1>       mov   al, [audio_device]
5813 00013900 3C03                <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
5814 00013902 7418                <1>       je    short snd_disable_1
5815 00013904 0F8711FFFFFF        <1>       ja    snd_nothing ; temporary.
5816 0001390A 3C01                <1>       cmp   al, 1 ; Sound Blaster 16
5817 0001390C 7507                <1>       jne   short snd_disable_0
5818 0001390E E8A01B0000          <1>       call  sb16_stop
5819 00013913 EB0C                <1>       jmp   short snd_disable_2
5820                              <1> snd_disable_0:
5821 00013915 E8CE1E0000          <1>       call  ac97_stop
5822 0001391A EB05                <1>       jmp   short snd_disable_2
5823                              <1> snd_disable_1:
5824 0001391C E85D170000          <1>       call  vt8233_stop
5825                              <1> snd_disable_2:
5826 00013921 A0[739B0100]        <1>       mov   al, [audio_intr]
5827 00013926 29DB                <1>       sub   ebx, ebx ; 0 = reset
5828 00013928 E851FAFFFF          <1>       call  set_dev_IRQ_service
5829                              <1>
5830                              <1>       ;mov  al, [audio_intr]
5831 0001392D 28E4                <1>       sub   ah, ah ; 0 = reset
5832 0001392F E8B5F6FFFF          <1>       call  set_hardware_int_vector
5833                              <1>
5834 00013934 31C0                <1>       xor   eax, eax
5835 00013936 A2[719B0100]        <1>       mov   byte [audio_device], al
5836 0001393B A2[739B0100]        <1>       mov   byte [audio_intr], al
5837 00013940 8705[909B0100]      <1>       xchg  eax, [audio_dma_buff]
5838                              <1>       ; 24/05/2017
5839                              <1>       ;or   eax, eax
5840                              <1>       ;jz   short snd_disable_3
5841                              <1>       ;cmp  eax, sb16_dma_buffer ; default DMA buffer
5842                              <1>       ;je   short snd_disable_3
5843 00013946 803D[709B0100]00    <1>       cmp   byte [audio_pci], 0 ; AC97 audio controller ?
5844 0001394D 7612                <1>       jna   short snd_disable_3
5845 0001394F C605[709B0100]00    <1>       mov   byte [audio_pci], 0
5846                              <1>       ;sub  ecx, ecx
5847                              <1>       ;xchg ecx, [audio_dmabuff_size]
5848 00013956 8B0D[949B0100]      <1>       mov   ecx, [audio_dmabuff_size]
5849 0001395C E8FD2BFFFF          <1>       call  deallocate_memory_block
5850                              <1> snd_disable_3:
5851 00013961 C3                  <1>       retn
5852                              <1>
5853                              <1> sound_dma_map:
5854                              <1>       ; FUNCTION = 13
5855                              <1>       ; Map audio dma buff addr to user's buffer addr
5856                              <1>       ; 12/05/2017
5857 00013962 21C9                <1>       and   ecx, ecx
5858 00013964 0F8408FBFFFF        <1>       jz    sound_buff_error
5859 0001396A 803D[719B0100]01    <1>       cmp   byte [audio_device], 1
5860 00013971 7229                <1>       jb    short snd_dma_map_1
5861                              <1> snd_dma_map_0:
5862 00013973 A1[909B0100]        <1>       mov   eax, [audio_dma_buff]
5863 00013978 21C0                <1>       and   eax, eax
5864 0001397A 7420                <1>       jz    short snd_dma_map_1
5865                              <1>       ;
5866 0001397C 8A1D[999B0100]      <1>       mov   bl, [audio_user]
5867 00013982 08DB                <1>       or    bl, bl
5868 00013984 7416                <1>       jz    short snd_dma_map_1
5869 00013986 3A1D[B3030300]      <1>       cmp   bl, [u.uno]
5870 0001398C 0F852BFCFFFF        <1>       jne   sndc_owner_error
5871                              <1>       ;
5872 00013992 8B1D[949B0100]      <1>       mov   ebx, [audio_dmabuff_size]
5873 00013998 21DB                <1>       and   ebx, ebx
5874 0001399A 750A                <1>       jnz   short snd_dma_map_2
5875                              <1> snd_dma_map_1:
5876 0001399C B8[00000200]        <1>       mov   eax, sb16_dma_buffer
5877 000139A1 BB00000100          <1>       mov   ebx, 65536
5878                              <1> snd_dma_map_2:
5879 000139A6 81C1FF0F0000        <1>       add   ecx, PAGE_SIZE-1 ; 4095
5880 000139AC 6681E100F0          <1>       and   cx, ~PAGE_OFF ; not 4095
5881 000139B1 39D9                <1>       cmp   ecx, ebx
5882 000139B3 0F87B9FAFFFF        <1>       ja    sound_buff_error
5883 000139B9 50                  <1>       push  eax
5884 000139BA 89D3                <1>       mov   ebx, edx
5885 000139BC C1E90C              <1>       shr   ecx, 12 ; byte count to page count
5886                              <1>       ; eax = physical address of (audio) dma buffer
5887                              <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
5888                              <1>       ; ecx = page count (>0)
5889 000139BF E80A2CFFFF          <1>       call  direct_memory_access
5890 000139C4 58                  <1>       pop   eax
5891 000139C5 0F82A7FAFFFF        <1>       jc    sound_buff_error
5892 000139CB A3[64030300]        <1>       mov   [u.r0], eax
5893 000139D0 C3                  <1>       retn
5894                              <1>
5895                              <1> soundc_info:
5896                              <1>       ; FUNCTION = 14
```

```
                            <1>       ; Get Audio Controller Info
                            <1>       ; 10/06/2017
                            <1>       ; 05/06/2017
000139D1 20DB               <1>       and   bl, bl ; 0
000139D3 740A               <1>       jz    short sndc_info_0
                            <1>       ; invalid parameter !
000139D5 B817000000         <1>       mov   eax, ERR_INV_PARAMETER ; 23
                            <1> ;sndc_inf_error:
                            <1> ;     mov   [u.r0], eax
                            <1> ;     mov   [u.error], eax
                            <1> ;     jmp   error
000139DA E99FFAFFFF         <1>       jmp   sysaudio_err
                            <1>
                            <1> sndc_info_0:
000139DF E8D4000000         <1>       call  snd_dev_check
000139E4 0F8281FAFFFF       <1>       jc    soundc_dev_err
                            <1>
000139EA 8B1D[7C9B0100]     <1>       mov   ebx, [audio_vendor]
000139F0 8B0D[789B0100]     <1>       mov   ecx, [audio_dev_id]
                            <1>       ;mov  al,  [audio_device]
000139F6 3C02               <1>       cmp   al, 2 ; AC'97 (ICH)
000139F8 7513               <1>       jne   short sndc_info_1
                            <1>       ; Intel AC97 (ICH) Audio Controller (=2)
000139FA 668B15[AA9B0100]   <1>       mov   dx, [NABMBAR]
00013A01 C1E210             <1>       shl   edx, 16
00013A04 668B15[A89B0100]   <1>       mov   dx, [NAMBAR]
00013A0B EB07               <1>       jmp   short sndc_info_2
                            <1> sndc_info_1:
                            <1>       ; 05/06/2017
                            <1>       ; Note: Intel HDA code (here) is not ready yet!
                            <1>       ; !!! SB16 or VT8233 (VT8237R) !!!
00013A0D 0FB715[769B0100]   <1>       movzx  edx, word [audio_io_base]
                            <1> sndc_info_2:
00013A14 88C4               <1>       mov   ah, al ;  [audio_device]
00013A16 A0[739B0100]       <1>       mov   al, [audio_intr]
                            <1>
                            <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>
                            <1>       ; 10/06/2017
00013A1B A3[64030300]       <1>       mov   [u.r0], eax
00013A20 8B2D[60030300]     <1>       mov   ebp, [u.usp]
00013A26 895D10             <1>       mov   [ebp+16], ebx  ; ebx
00013A29 895514             <1>       mov   [ebp+20], edx  ; edx
00013A2C 894D18             <1>       mov   [ebp+24], ecx  ; ecx
                            <1>
00013A2F C3                 <1>       retn
                            <1>
                            <1> sound_data:
                            <1>       ; FUNCTION = 15
                            <1>       ; Get Current Sound data for graphics
                            <1>       ; 22/06/2017
                            <1>       ;
00013A30 E883000000         <1>       call  snd_dev_check
00013A35 0F8230FAFFFF       <1>       jc    soundc_dev_err ; Device not ready !
                            <1>
00013A3B 80FB00             <1>       cmp   bl, 0
00013A3E 760A               <1>       jna   short sound_data_0
                            <1>
                            <1>       ; Only PCM OUT buffer data is valid for now!
00013A40 B817000000         <1>       mov   eax, ERR_INV_PARAMETER  ; 23
00013A45 E934FAFFFF         <1>       jmp   sysaudio_err
                            <1>
                            <1> sound_data_0:
00013A4A A1[909B0100]       <1>       mov   eax, [audio_dma_buff]
00013A4F 09C0               <1>       or    eax, eax
00013A51 0F841BFAFFFF       <1>       jz    sound_buff_error
                            <1>
00013A57 803D[719B0100]04   <1>       cmp   byte [audio_device], 4 ; Intel HDA
00013A5E 744F               <1>       je    short sound_data_4 ; temporary ! (22/06/2017)
                            <1>
00013A60 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>
00013A62 0F84771F0000       <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>
00013A68 3B0D[949B0100]     <1>       cmp   ecx, [audio_dmabuff_size]
00013A6E 0F87FEF9FFFF       <1>       ja    sound_buff_error
                            <1>
00013A74 89D0               <1>       mov   eax, edx
00013A76 25FF0F0000         <1>       and   eax, PAGE_OFF ; 4095 (0FFFh)
```

```
6002 00013A7B 81F900100000        <1>       cmp    ecx, 4096
6003 00013A81 7605                <1>       jna    short sound_data_2
6004 00013A83 B900100000          <1>       mov    ecx, 4096 ; max. 1 page
6005                              <1> sound_data_2:
6006 00013A88 01C8                <1>       add    eax, ecx
6007 00013A8A 3D00100000          <1>       cmp    eax, 4096
6008 00013A8F 7606                <1>       jna    short sound_data_3
6009 00013A91 6625FF0F            <1>       and    ax, PAGE_OFF ; 4095 (0FFFh)
6010 00013A95 29C1                <1>       sub    ecx, eax
6011                              <1>       ; here, ECX has been adjusted to fit
6012                              <1>       ;      in page border..  (<= 4096, >0)
6013                              <1> sound_data_3:
6014 00013A97 51                  <1>       push   ecx
6015 00013A98 52                  <1>       push   edx
6016 00013A99 89D3                <1>       mov    ebx, edx
6017 00013A9B E81C27FFFF          <1>       call   get_physical_addr
6018 00013AA0 5A                  <1>       pop    edx
6019 00013AA1 59                  <1>       pop    ecx
6020 00013AA2 0F82CAF9FFFF        <1>       jc     sound_buff_error
6021                              <1>
6022                              <1>       ; eax = physical address of user's buffer
6023 00013AA8 89C3                <1>       mov    ebx, eax
6024                              <1>       ; ecx = byte (transfer) count
6025                              <1>       ;call  get_current_sound_data
6026                              <1>       ;retn
6027 00013AAA E98D1E0000          <1>       jmp    get_current_sound_data
6028                              <1>
6029                              <1> sound_data_4:
6030                              <1>       ; Intel HDA code is not ready yet !
6031                              <1>       ; 22/06/2017
6032 00013AAF 31C0                <1>       xor    eax, eax
6033 00013AB1 48                  <1>       dec    eax
6034 00013AB2 A3[64030300]        <1>       mov    [u.r0], eax ; 0FFFFFFFFh
6035 00013AB7 C3                  <1>       retn
6036                              <1>
6037                              <1> snd_dev_check:
6038                              <1>       ; 10/06/2017
6039                              <1>       ; 05/06/2017
6040                              <1>       ; 24/05/2017
6041                              <1>       ; 22/04/2017
6042                              <1>       ; 21/04/2017
6043                              <1>       ; ... device check at first
6044 00013AB8 A0[719B0100]        <1>       mov    al, [audio_device]
6045 00013ABD 3C01                <1>       cmp    al, 1 ; SB 16
6046 00013ABF 7203                <1>       jb     short snd_dev_chk_retn ; error !
6047                              <1>       ;cmp   al, 4 ; Intel HDA
6048                              <1>       ;ja    short snd_dbchk_stc ; invalid !
6049                              <1>       ; 10/06/2017
6050 00013AC1 3C05                <1>       cmp    al, 5
6051 00013AC3 F5                  <1>       cmc
6052                              <1> snd_dev_chk_retn:
6053 00013AC4 C3                  <1>       retn
6054                              <1>
6055                              <1> snd_buf_check:
6056                              <1>       ; 10/06/2017
6057                              <1>       ; 22/04/2017
6058                              <1>       ; 21/04/2017
6059                              <1>       ; ... buffer & (buffer) owner check at second
6060 00013AC5 833D[849B0100]00    <1>       cmp    dword [audio_buffer], 0
6061 00013ACC 760D                <1>       jna    short snd_dbchk_stc
6062                              <1> snd_user_check:
6063 00013ACE A0[B3030300]        <1>       mov    al, [u.uno]
6064 00013AD3 3A05[999B0100]      <1>       cmp    al, [audio_user]
6065                              <1>       ;jne   short snd_dbchk_stc
6066                              <1>       ;retn
6067 00013AD9 74E9                <1>       je     short snd_dev_chk_retn
6068                              <1>
6069                              <1> snd_dbchk_stc:
6070 00013ADB F9                  <1>       stc
6071 00013ADC C3                  <1>       retn
6072                              <1>
6073                              <1> sound_update:
6074                              <1>       ; FUNCTION = 16
6075                              <1>       ; bl =
6076                              <1>       ;    0 = automatic (sequental) update (with flag switch!)
6077                              <1>       ;    1 = update dma half buffer 1 (without flag switch!)
6078                              <1>       ;    2 = update dma half buffer 2 (without flag switch!)
6079                              <1>       ;  FFh = get current flag value
6080                              <1>       ;       0 = dma half buffer 1 (will be played next)
6081                              <1>       ;       1 = dma half buffer 2 (will be played next)
6082                              <1>
6083                              <1>       ; 10/10/2017
6084                              <1>
6085                              <1>       ; ... device check at first
6086 00013ADD A0[719B0100]        <1>       mov    al, [audio_device]
6087 00013AE2 08C0                <1>       or     al, al ; 0 ; pc speaker or invalid
6088 00013AE4 0F8481F9FFFF        <1>       jz     soundc_dev_err
6089                              <1>
6090                              <1>       ; ... buffer & (buffer) owner check at second
6091 00013AEA 833D[849B0100]00    <1>       cmp    dword [audio_buffer], 0
6092 00013AF1 0F867BF9FFFF        <1>       jna    sound_buff_error
6093 00013AF7 A0[B3030300]        <1>       mov    al, [u.uno]
6094 00013AFC 3A05[999B0100]      <1>       cmp    al, [audio_user]
6095 00013B02 0F85B5FAFFFF        <1>       jne    sndc_owner_error
6096                              <1>
6097                              <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
6098 00013B08 8B3D[909B0100]      <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
6099 00013B0E 09FF                <1>       or     edi, edi
6100 00013B10 0F845CF9FFFF        <1>       jz     sound_buff_error
6101 00013B16 8B35[889B0100]      <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
6102 00013B1C 8B0D[8C9B0100]      <1>       mov    ecx, [audio_buff_size]
6103                              <1>
6104                              <1>       ;movzx eax, byte [audio_flag]
6105 00013B22 A0[989B0100]        <1>       mov    al, [audio_flag]
6106                              <1>
```

```
6107 00013B27 FEC3                <1>       inc    bl
6108 00013B29 7427                <1>       jz     short snd_update_3 ; bl = 0FFh
6109 00013B2B FECB                <1>       dec    bl
6110 00013B2D 7411                <1>       jz     short snd_update_0 ; bl = 0
6111                              <1>
6112 00013B2F 80FB02              <1>       cmp    bl, 2
6113 00013B32 7417                <1>       je     short snd_update_1 ; dma half buffer 2
6114 00013B34 7217                <1>       jb     short snd_update_2 ; dma half buffer 1
6115                              <1>
6116                              <1>       ; invalid parameter !
6117 00013B36 B817000000          <1>       mov    eax, ERR_INV_PARAMETER ; 23
6118                              <1> ;     mov    [u.r0], eax
6119                              <1> ;     mov    [u.error], eax
6120                              <1> ;     jmp    error
6121 00013B3B E93EF9FFFF          <1>       jmp    sysaudio_err
6122                              <1>
6123                              <1> snd_update_0:
6124 00013B40 8035[989B0100]01    <1>       xor    byte [audio_flag], 1 ; update flag !!!
6125 00013B47 3C01                <1>       cmp    al, 1
6126 00013B49 7202                <1>       jb     short snd_update_2 ; dma half buffer 1
6127                              <1> snd_update_1:
6128                              <1>       ; dma half buffer 2
6129 00013B4B 01CF                <1>       add    edi, ecx
6130                              <1> snd_update_2:
6131                              <1>       ;rep   movsb
6132 00013B4D C1E902              <1>       shr    ecx, 2
6133 00013B50 F3A5                <1>       rep    movsd
6134                              <1> snd_update_3:
6135 00013B52 A3[64030300]        <1>       mov    [u.r0], eax
6136                              <1>
6137 00013B57 C3                  <1>       retn
6138                              <1>
6139                              <1> set_irq_callback_service:
6140                              <1>       ; 03/08/2020
6141                              <1>       ; 10/06/2017
6142                              <1>       ; 12/05/2017
6143                              <1>       ; 24/04/2017
6144                              <1>       ; 22/04/2017
6145                              <1>       ; caller: 'syscalbac' or 'sysaudio' or ...
6146                              <1>       ; 13/04/2017, 14/04/2017, 17/04/2017
6147                              <1>       ; 24/02/2017, 26/02/2017, 28/02/2017
6148                              <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
6149                              <1>       ;
6150                              <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
6151                              <1>       ;
6152                              <1>       ; INPUT ->
6153                              <1>       ;     If AL = 0, the caller is 'syscalbac';
6154                              <1>       ;        otherwise, the caller is 'sysaudio' or ...
6155                              <1>       ;        (AL = user number)
6156                              <1>       ;
6157                              <1>       ;     BL = IRQ number (Hardware interrupt request number)
6158                              <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
6159                              <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
6160                              <1>       ;          (numbers >15 are invalid)
6161                              <1>       ;
6162                              <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
6163                              <1>       ;          1 = Link IRQ by using Signal Response Byte method
6164                              <1>       ;          2 = Link IRQ by using Callback service method
6165                              <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
6166                              <1>       ;         >3 = invalid
6167                              <1>       ;             (syscallback version will return to user)
6168                              <1>       ;
6169                              <1>       ;     CL = Signal Return/Response Byte value
6170                              <1>       ;
6171                              <1>       ;     If BH = 3, kernel will put a counter value ; 03/08/2020
6172                              <1>       ;               (into the S.R.B. addr)
6173                              <1>       ;              between 0 to 255. (start value = CL+1)
6174                              <1>       ;
6175                              <1>       ;     NOTE: counter value, for example: even and odd numbers
6176                              <1>       ;           may be used for -audio- DMA buffer switch
6177                              <1>       ;           within double buffer method, etc.
6178                              <1>       ;
6179                              <1>       ;     EDX = Signal return (Response) byte address
6180                              <1>       ;                           - or -
6181                              <1>       ;           Interrupt/Callback service/routine address
6182                              <1>       ;
6183                              <1>       ;           (virtual address in user's memory space)
6184                              <1>       ;
6185                              <1>       ; OUTPUT ->
6186                              <1>       ;     CF = 0 & EAX = 0 -> Successful setting
6187                              <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
6188                              <1>       ;                         by another process
6189                              <1>       ;             eax = ERR_PERM_DENIED -> prohibited or locked
6190                              <1>       ;             eax = ERR_INV_PARAMETER ->
6191                              <1>       ;                   invalid parameter/option or bad address
6192                              <1>       ;
6193                              <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
6194                              <1>       ;
6195                              <1>       ;        [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
6196                              <1>       ;                      which IRQs are locked by (that) user.
6197                              <1>       ;                       Lock and unlock (by user) will change
6198                              <1>       ;                      these flags or 'terminate process' (sysexit)
6199                              <1>       ;                      will clear these flags and unlock those IRQs.
6200                              <1>       ;
6201                              <1>       ;                      Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
6202                              <1>       ;
6203                              <1>       ;        IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
6204                              <1>       ;
6205                              <1>       ;        IRQ(x).method : 1 byte for callback method & status
6206                              <1>       ;                        0 = Signal Response Byte method
6207                              <1>       ;                        1 = Callback service method
6208                              <1>       ;                        >1 = invalid for current 'syscalback'.
6209                              <1>       ;                      or(+) 80h = IRQ is in use by system (ring 0)
6210                              <1>       ;                                  function (audio etc.) or
6211                              <1>       ;                                a device driver.
```

```
6212                                    <1>     ;                   (system function will ignore the lock/owner)
6213                                    <1>     ;
6214                                    <1>     ;       IRQ(x).srb: 1 byte, Signal Return/Response byte value
6215                                    <1>     ;                    (a fixed value by user or a counter value
6216                                    <1>     ;                   from 0 to 255, which is increased by every
6217                                    <1>     ;                   interrupt just before putting it into
6218                                    <1>     ;                   the Signal Response byte address
6219                                    <1>     ;                   (This is not used in callback serv method)
6220                                    <1>     ;
6221                                    <1>     ;       IRQ(x).addr     : 1 dword
6222                                    <1>     ;                    Signal Response Byte address (physical)
6223                                    <1>     ;                               -or-
6224                                    <1>     ;                    Callback service address (virtual)
6225                                    <1>     ;
6226                                    <1>     ;       IRQ(x).dev: 1 byte
6227                                    <1>     ;                    0 = Default device or kernel function
6228                                    <1>     ;                               -or-
6229                                    <1>     ;                    1-255 = Assigned device driver number
6230                                    <1>     ;
6231                                    <1>     ;       (x) = 3,4,5,7,9,10,11,12,13
6232                                    <1>     ;
6233                                    <1>
6234 00013B58 80FB0F                   <1>     cmp   bl, 15
6235 00013B5B 7729                     <1>     ja    short scbs_2
6236                                    <1>
6237 00013B5D 80FF03                   <1>     cmp   bh, 3
6238 00013B60 7724                     <1>     ja    short scbs_2  ; invalid parameter
6239                                    <1>
6240 00013B62 0FB6FB                   <1>     movzx  edi, bl ; save IRQ number
6241                                    <1>
6242                                    <1>           ;  IRQ 0,1,2,6,8,14,15 are prohibited
6243                                    <1>     ;IRQenum: ; 'trdosk9.s'
6244                                    <1>     ;     db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
6245                                    <1>
6246 00013B65 0FB6B7[B8480100]         <1>     movzx  esi, byte [edi+IRQenum] ; IRQ availability
6247                                    <1>                                    ; enumeration/index
6248                                    <1>     ;dec   esi
6249 00013B6C 664E                     <1>     dec   si
6250 00013B6E 780F                     <1>     js    short scbs_1 ;  0 -> 0FFFFh
6251                                    <1>
6252                                    <1>     ; ESI = IRQ callback parameters index number (0 to 8)
6253                                    <1>
6254 00013B70 08FF                     <1>     or    bh, bh
6255 00013B72 7419                     <1>     jz    short scbs_4 ; unlink the IRQ (in BL)
6256                                    <1>
6257 00013B74 FECF                     <1>     dec   bh
6258                                    <1>     ; bh = method (0 = signal response byte, 1 = callback)
6259                                    <1>     ;           (2 = auto increment of signal response byte)
6260                                    <1>
6261 00013B76 80BE[229B0100]00         <1>     cmp   byte [esi+IRQ.owner], 0 ; locked ?
6262 00013B7D 7637                     <1>     jna   short scbs_6 ; no... OK...
6263                                    <1>
6264                                    <1> scbs_1:
6265                                    <1>     ; permission denied (prohibited IRQ)
6266 00013B7F B80B000000               <1>     mov   eax, ERR_PERM_DENIED
6267 00013B84 F9                       <1>     stc
6268 00013B85 C3                       <1>     retn
6269                                    <1> scbs_2:
6270 00013B86 F9                       <1>     stc
6271                                    <1> scbs_3:
6272 00013B87 B817000000               <1>     mov   eax, ERR_INV_PARAMETER
6273 00013B8C C3                       <1>     retn
6274                                    <1>
6275                                    <1> scbs_4: ; unlink the requested IRQ (if it belongs to current user)
6276                                    <1>     ; 10/06/2017
6277                                    <1>     ; 22/04/2017
6278                                    <1>     ; 14/04/2017
6279                                    <1>     ; If AL = 0 -> The caller is 'syscalbac'
6280 00013B8D 8AA6[229B0100]           <1>     mov   ah, [esi+IRQ.owner]
6281 00013B93 3A25[B3030300]           <1>     cmp   ah, [u.uno]
6282 00013B99 75E4                     <1>     jne   short scbs_1
6283                                    <1>
6284 00013B9B FE0D[D6030300]           <1>     dec   byte [u.irqc] ; decrease IRQ count (in use)
6285                                    <1>
6286                                    <1>     ;sub   ah, ah
6287                                    <1>     ;mov   [esi+IRQ.owner], ah ; 0 ; free !!!
6288                                    <1>     ;and   byte [esi+IRQ.method], 80h
6289                                    <1>     ;mov   [esi+IRQ.srb], ah ; 0
6290                                    <1>     ;mov   [esi+IRQ.dev], ah ; 0
6291                                    <1>     ;mov   dword [esi+IRQ.addr], 0
6292                                    <1>     ;mov   dword [u.r0], 0
6293                                    <1>
6294                                    <1>     ;mov   byte [esi+IRQ.owner], 0
6295                                    <1>
6296                                    <1>     ; 22/04/2017
6297 00013BA1 29C0                     <1>     sub   eax, eax
6298 00013BA3 8886[229B0100]           <1>     mov   [esi+IRQ.owner], al ; 0
6299                                    <1>     ; 10/06/2017
6300 00013BA9 8686[349B0100]           <1>     xchg  al, [esi+IRQ.method]
6301 00013BAF 2480                     <1>     and   al, 80h
6302 00013BB1 745E                     <1>     jz    short scbs_12
6303                                    <1>     ; Audio device must be disabled -later- ! ([IRQ.medhod] = 80h)
6304                                    <1>
6305                                    <1> ;     cmp   byte [esi+IRQ.method], 80h ; device drv or kernel extension ?
6306                                    <1> ;     jb    short scbs_12 ; bh = 0 reset to default IRQ handler
6307                                    <1> ;
6308                                    <1> ;     and   al, al
6309                                    <1> ;     jz    short  scbs_5  ; the caller is 'syscalbac'
6310                                    <1> ;     ; The caller is 'sysaudio' or ...
6311 00013BB3 30C0                     <1>     xor   al, al
6312                                    <1> ;     mov   [esi+IRQ.method], al ; 0 ; reset kernel extension flag
6313                                    <1> ;scbs_5:
6314                                    <1> ;     sub   ah, ah
6315                                    <1>     ;mov   [u.r0], eax ; 0
6316 00013BB5 C3                       <1>     retn
```

```
                                <1>
                                <1> scbs_6:
                                <1>       ; 14/04/2017
00013BB6 20C0                   <1>       and   al, al
00013BB8 7405                   <1>       jz    short scbs_7  ; the caller is 'syscalbac'
                                <1>       ; AL = user number ([u.uno] or [audio.user] or ...)
                                <1>       ; The caller is 'sysaudio' or ...
                                <1>       ;
                                <1>       ; bh = method (0 = signal response byte, 1 = callback)
                                <1>       ;            (2 = auto increment of signal response byte)
                                <1>
00013BBA 80CF80                 <1>       or    bh, 80h             ; Kernel extension flag !
00013BBD EB0A                   <1>       jmp   short scbs_8
                                <1> scbs_7:
00013BBF 8A86[349B0100]         <1>       mov   al, [esi+IRQ.method] ; >= 80h = kernel is using this IRQ
00013BC5 2480                   <1>       and   al, 80h ; use only bit 7 (kernel function flag)
00013BC7 08C7                   <1>       or    bh, al        ; method
                                <1>                           ; 0 = signal response byte, 1 = callback
                                <1>                           ; 2 = auto increment of s.r.b.
                                <1> scbs_8:
00013BC9 A0[B3030300]           <1>       mov   al, [u.uno] ; user (process) number (1 to 16)
00013BCE 8886[229B0100]         <1>       mov   [esi+IRQ.owner], al ; lock the IRQ for user
00013BD4 88BE[349B0100]         <1>       mov   [esi+IRQ.method], bh
                                <1>
                                <1> ;     test  bh, 1
                                <1> ;     jnz   short scbs_9  ; Callback method, CX will not be used
                                <1> ;
                                <1> ;     test  bh, 2         ; use auto increment (counter) method
                                <1> ;     jz    short scbs_10 ; (count can be used for buffer switch)
                                <1> ;scbs_9:
                                <1> ;     xor   ecx, ecx ; 0
                                <1> scbs_10:
                                <1>       ;mov  [esi+IRQ.method], bh
00013BDA 888E[3D9B0100]         <1>       mov   [esi+IRQ.srb], cl
00013BE0 C686[2B9B0100]00       <1>       mov   byte [esi+IRQ.dev], 0 ; device number is always 0
                                <1>                             ; for this system call
                                <1>       ;test bh, 1
00013BE7 80E701                 <1>       and   bh, 1 ; 17/04/2017
00013BEA 7513                   <1>       jnz   short scbs_11 ; callback method, use virtual address
                                <1>
00013BEC 53                     <1>       push  ebx ; IRQ number (in BL)
00013BED 89D3                   <1>       mov   ebx, edx
                                <1>       ; ebx = virtual address
                                <1>       ; [u.pgdir] = page directory's physical address
00013BEF FE05[C29A0100]         <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.
00013BF5 E8C225FFFF             <1>       call  get_physical_addr
00013BFA 5B                     <1>       pop   ebx
00013BFB 728A                   <1>       jc    scbs_3 ; invalid address !
                                <1>       ; eax = physical address of the virtual address in user's space
00013BFD 89C2                   <1>       mov   edx, eax
                                <1> scbs_11:
00013BFF 66C1E602               <1>       shl   si, 2        ; byte (index) to dword (offset)
00013C03 8996[469B0100]         <1>       mov   [esi+IRQ.addr], edx
                                <1>
00013C09 FE05[D6030300]         <1>       inc   byte [u.irqc]; increase IRQ (in use) count
                                <1>
00013C0F FEC7                   <1>       inc    bh ; 17/04/2017
                                <1>       ; bh > 0 -> set to requested IRQ handler (IRQ_u_list)
                                <1> scbs_12:
00013C11 88D8                   <1>       mov   al, bl ; IRQ number
00013C13 88FC                   <1>       mov   ah, bh ; 0 = reset, >0 = set
00013C15 E8CFF3FFFF             <1>       call  set_hardware_int_vector
                                <1>
00013C1A 31C0                   <1>       xor   eax, eax
                                <1>       ;mov  [u.r0], eax ; 0
                                <1>
00013C1C C3                     <1>       retn  ; return with success (cf=0, eax=0)
                                <1>
                                <1>
                                <1> sysdma: ; DMA FUNCTIONS
                                <1>       ; 02/09/2017
                                <1>       ; 28/08/2017
                                <1>       ; 20/08/2017 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; Inputs:
                                <1>       ;     BH = 0 -> Allocate DMA buffer
                                <1>       ;        BL = 0 -> Use the system's default DMA
                                <1>       ;                  (SB16) Buffer
                                <1>       ;              Buffer Size (max.) = 65536 bytes
                                <1>       ;        BL > 0 -> Allocate (a new) DMA buffer
                                <1>       ;        ECX = DMA Buffer Size in bytes (<=128KB)
                                <1>       ;        EDX = Virtual Address of DMA buffer
                                <1>       ;
                                <1>       ;     BH = 1 -> Initialize (Start) DMA service
                                <1>       ;          BL, bit 0 to 3 = Channel Number (0 to 7)
                                <1>       ;          BL, bit 7 = Auto Initialized Mode
                                <1>       ;                  (If bit 7 is set)
                                <1>       ;             bit 6 = Record (read) mode (0= playback)
                                <1>       ;          ECX = byte count (0 = use dma buffer size)
                                <1>       ;          EDX = physical buffer address
                                <1>       ;               (0 = use dma buffer -start- address)
                                <1>       ;
                                <1>       ;     BH = 2 -> Get Current DMA Buffer Offset
                                <1>       ;          BL = DMA channel number
                                <1>       ;
                                <1>       ;     BH = 3 -> Get Current DMA count down value
                                <1>       ;          BL = DMA channel number (0 tO 7)
                                <1>       ;
                                <1>       ;     BH = 4 -> Get Current DMA channel (in progress)
                                <1>       ;
                                <1>       ;     BH = 5 -> Get System's Default DMA Buffer Address
                                <1>       ;
```

```
                                        <1>       ;      BH = 6 -> Get Current DMA Buffer Address
                                        <1>       ;
                                        <1>       ;      BH = 7 -> Stop DMA service
                                        <1>       ;
                                        <1>       ;
                                        <1>       ; Outputs:
                                        <1>       ;
                                        <1>       ;      For BH = 0 ; Allocate DMA buffer
                                        <1>       ;          EAX = Physical address of DMA buffer
                                        <1>       ;          ECX = Allocated buffer size in bytes
                                        <1>       ;               - page count * 4096 -
                                        <1>       ;                (may be bigger than requested)
                                        <1>       ;          If BL input > 0,
                                        <1>       ;             'sysalloc:' system call will be used with
                                        <1>       ;             EBX (for 'sysalloc') = EDX (for 'sysdma')
                                        <1>       ;             ECX is same, byte count (buffer size)
                                        <1>       ;             EDX = 1024*1024*16 ; 16 MB upper limit
                                        <1>       ;          If BL input = 0,
                                        <1>       ;             Default DMA buffer (SB16 buffer) will be
                                        <1>       ;             checked and if it is free, it's address
                                        <1>       ;             will be returned in EAX and it's size
                                        <1>       ;             will be returned in ECX (as 65536)
                                        <1>       ;
                                        <1>       ;          If CF = 1, error code is in EAX
                                        <1>       ;             EAX = -1 ; DMA buffer allocation error!
                                        <1>       ;             EAX = 11 ; 'Permission Denied' error !
                                        <1>       ;
                                        <1>       ;             Note: 'sysalloc' error return method
                                        <1>       ;                   will be applied if BL input > 0 !
                                        <1>       ;
                                        <1>       ;       For BH = 1 ; Initialize (Start) DMA
                                        <1>       ;           EAX = 0 (Successful)
                                        <1>       ;           If CF = 1, error code is in EAX
                                        <1>       ;
                                        <1>       ;       For BH = 2 ; Get Current DMA Buffer Offset
                                        <1>       ;           EAX = DMA Buffer Offset (in bytes)
                                        <1>       ;           ;
                                        <1>       ;            AX = DMA buffer offset
                                        <1>       ;           EAX bits 16 to 23 = Page register value
                                        <1>       ;
                                        <1>       ;       For BH = 3 ; Get Current DMA count down value
                                        <1>       ;           EAX = Count down value (remain bytes)
                                        <1>       ;
                                        <1>       ;       For BH = 4 ; Get Current DMA channel (in progress)
                                        <1>       ;           EAX = DMA channel number (0 to 7)
                                        <1>       ;             AH = 0 if the owner is the caller process
                                        <1>       ;             AH > 0 if the dma channel is in use by
                                        <1>       ;                    another user/process
                                        <1>       ;           EAX = -1 (0FFFFFFFFh)
                                        <1>       ;                if DMA service is not in use
                                        <1>       ;                (stopped or not initialized/started)
                                        <1>       ;
                                        <1>       ;       For BH = 5 ; Get System's Default DMA Buff Addr
                                        <1>       ;           EAX = Default DMA Buffer Address (Physical)
                                        <1>       ;             = offset 'sb16_dma_buffer:'
                                        <1>       ;           ECX = Buffer size
                                        <1>       ;             = 65536
                                        <1>       ;
                                        <1>       ;       For BH = 6 ; Get Current DMA Buffer Address
                                        <1>       ;           EAX = Current DMA buffer address (Physical)
                                        <1>       ;           ECX = Current DMA buffer size (setting value)
                                        <1>       ;           Note: These values are for current dma channel
                                        <1>       ;                settings for the user/process
                                        <1>       ;           ** For now (for current TRDOS 386 version)
                                        <1>       ;             only one user/process can use only one
                                        <1>       ;             dma channel & one dma buffer at same time
                                        <1>       ;             (no multi tasking on DMA service) !!! **
                                        <1>       ;           (Once, current DMA user must stop it's own DMA
                                        <1>       ;            DMA service, than another user/program
                                        <1>       ;            can use DMA service with same dma channel
                                        <1>       ;            or with another DMA channel.)
                                        <1>       ;
                                        <1>       ;       For BH = 7 ; Stop DMA service (for current user
                                        <1>       ;           and current DMA channel)
                                        <1>       ;           EAX = 0 ; successful
                                        <1>       ;           CF = 1 & EAX > 0 (= -1) -> Error
                                        <1>
00013C1D 80FF07                         <1>       cmp    bh, 7
00013C20 7612                           <1>       jna    short sysdma_0
                                        <1>
                                        <1> sysdma_err:
00013C22 31C0                           <1>       xor    eax, eax
00013C24 48                             <1>       dec    eax ; -1
                                        <1> sysdma_perm_err:
00013C25 A3[64030300]                   <1>       mov    [u.r0], eax
00013C2A A3[C8030300]                   <1>       mov    [u.error], eax ; DMA service error !
00013C2F E9789BFFFF                     <1>       jmp    error
                                        <1>
                                        <1> sysdma_0:
00013C34 08FF                           <1>       or     bh, bh
00013C36 0F85BA000000                   <1>       jnz    sysdma_1
                                        <1>
00013C3C 20DB                           <1>       and    bl, bl
00013C3E 7416                           <1>       jz     short sysdma_01
                                        <1>
                                        <1>       ; redirect system call to 'sysalloc'
00013C40 89D3                           <1>       mov    ebx, edx ; virtual address of DMA buffer
                                        <1>       ;ecx = Buffer size in bytes
                                        <1>       ; DMA buffer address <= 16MB upper limit
00013C42 BA00000001                     <1>       mov    edx, 1024*1024*16 ; 16MB limit for DMA buff
                                        <1>
00013C47 C705[B49F0100]FFFF-            <1>       mov    dword [dma_addr], 0FFFFFFFFh ; -1
00013C4F FFFF                           <1>
                                        <1>
00013C51 E9A3E5FFFF                     <1>       jmp    sysalloc
```

```
6526                                    <1>
6527                                    <1> sysdma_01:
6528 00013C56 B8[00000200]              <1>         mov     eax, sb16_dma_buffer
6529                                    <1>
6530 00013C5B 803D[719B0100]01          <1>         cmp     byte [audio_device], 1
6531 00013C62 722A                      <1>         jb      short sysdma_03
6532                                    <1>
6533 00013C64 3B05[909B0100]            <1>         cmp     eax, [audio_dma_buff]
6534 00013C6A 7507                      <1>         jne     short sysdma_02
6535                                    <1>
6536                                    <1> sysdma_0_err:
6537 00013C6C B80B000000                <1>         mov     eax, ERR_PERM_DENIED
6538 00013C71 EBB2                      <1>         jmp     short sysdma_perm_err
6539                                    <1>
6540                                    <1> sysdma_02:
6541                                    <1>         ; Only one user is permitted for audio/dma functions
6542                                    <1>
6543 00013C73 833D[909B0100]00          <1>         cmp     dword [audio_dma_buff], 0
6544 00013C7A 7612                      <1>         jna     short sysdma_03
6545                                    <1>
6546 00013C7C 8A1D[999B0100]            <1>         mov     bl, [audio_user]
6547 00013C82 08DB                      <1>         or      bl, bl
6548 00013C84 7408                      <1>         jz      short sysdma_03
6549                                    <1>
6550 00013C86 3A1D[B3030300]            <1>         cmp     bl, [u.uno]
6551 00013C8C 75DE                      <1>         jne     short sysdma_0_err
6552                                    <1>
6553                                    <1> sysdma_03:
6554 00013C8E 8A1D[B19F0100]            <1>         mov     bl, [dma_user]
6555 00013C94 20DB                      <1>         and     bl, bl
6556 00013C96 750E                      <1>         jnz     short sysdma_04
6557                                    <1>
6558 00013C98 8A1D[B3030300]            <1>         mov     bl, [u.uno]
6559 00013C9E 881D[B19F0100]            <1>         mov     [dma_user], bl
6560                                    <1>
6561 00013CA4 EB15                      <1>         jmp     short sysdma_05
6562                                    <1>
6563                                    <1> sysdma_04:
6564 00013CA6 8B35[B49F0100]            <1>         mov     esi, [dma_addr]
6565 00013CAC 21F6                      <1>         and     esi, esi
6566 00013CAE 740B                      <1>         jz      short sysdma_05
6567                                    <1>
6568 00013CB0 46                        <1>         inc     esi ; -1 -> 0
6569 00013CB1 7408                      <1>         jz      short sysdma_05
6570                                    <1>
6571 00013CB3 3A1D[B3030300]            <1>         cmp     bl, [u.uno]
6572 00013CB9 75B1                      <1>         jne     short sysdma_0_err
6573                                    <1>
6574                                    <1> sysdma_05:
6575                                    <1>         ; edx = virtual address (user's buffer address)
6576                                    <1>         ;
6577 00013CBB 81F900000100              <1>         cmp     ecx, 65536   ; byte count (buffer size)
6578 00013CC1 0F875BFFFFFF              <1>         ja      sysdma_err
6579                                    <1>         ;
6580 00013CC7 81C1FF0F0000              <1>         add     ecx, PAGE_SIZE-1 ; 4095
6581 00013CCD 6681E100F0                <1>         and     cx, ~PAGE_OFF ; not 4095
6582                                    <1>         ;cmp    ecx, 65536
6583                                    <1>         ;ja     sysdma_err ;
6584 00013CD2 51                        <1>         push    ecx    ; buffer size (allocated pages * 4096)
6585 00013CD3 50                        <1>         push    eax    ; offset sb16_dma_buffer
6586 00013CD4 89D3                      <1>         mov     ebx, edx
6587 00013CD6 C1E90C                    <1>         shr     ecx, 12 ; byte count to page count
6588                                    <1>         ; eax = physical address of (audio) dma buffer
6589                                    <1>         ; ebx = virtual address of (audio) dma buffer (user's pgdir)
6590                                    <1>         ; ecx = page count (>0)
6591 00013CD9 E8F028FFFF                <1>         call    direct_memory_access
6592 00013CDE 58                        <1>         pop     eax
6593 00013CDF 59                        <1>         pop     ecx
6594 00013CE0 0F823CFFFFFF              <1>         jc      sysdma_err
6595                                    <1>
6596 00013CE6 A3[B49F0100]              <1>         mov     [dma_addr], eax
6597 00013CEB 890D[B89F0100]            <1>         mov     [dma_size], ecx ; dma buffer size (in bytes)
6598                                    <1>
6599                                    <1>         ;mov    [u.r0], eax ; DMA Buffer Address (Physical)
6600                                    <1>
6601                                    <1>         ;mov    ebp, [u.usp]  ; ebp points to user's registers
6602                                    <1>         ;mov    [ebp+24], ecx ; return to user with ecx value
6603                                    <1>
6604                                    <1>         ;jmp    sysret
6605                                    <1>
6606                                    <1>         ; 28/08/2017
6607 00013CF1 E9C4000000                <1>         jmp     sysdma_51
6608                                    <1>
6609                                    <1> sysdma_1:
6610 00013CF6 80FF01                    <1>         cmp     bh, 1
6611 00013CF9 0F87A6000000              <1>         ja      sysdma_5
6612                                    <1>
6613 00013CFF F6C340                    <1>         test    bl, 40h      ; record (read) mode -BL, bit 6-
6614 00013D02 0F851AFFFFFF              <1>         jnz     sysdma_err ; not ready yet!
6615                                    <1>
6616 00013D08 A1[B49F0100]              <1>         mov     eax, [dma_addr] ; physical address of dma buffer
6617 00013D0D 21C0                      <1>         and     eax, eax
6618 00013D0F 0F840DFFFFFF              <1>         jz      sysdma_err
6619                                    <1>
6620 00013D15 09D2                      <1>         or      edx, edx
6621 00013D17 7504                      <1>         jnz     short sysdma_11
6622                                    <1>
6623 00013D19 89C2                      <1>         mov     edx, eax
6624 00013D1B EB08                      <1>         jmp     short sysdma_12
6625                                    <1> sysdma_11:
6626 00013D1D 39C2                      <1>         cmp     edx, eax
6627 00013D1F 0F82FDFEFFFF              <1>         jb      sysdma_err
6628                                    <1> sysdma_12:
6629 00013D25 21C9                      <1>         and     ecx, ecx
6630 00013D27 7508                      <1>         jnz     short sysdma_13
```

```
6631                                <1>
6632 00013D29 8B0D[B89F0100]       <1>       mov   ecx, [dma_size]
6633 00013D2F EB0C                 <1>       jmp   short sysdma_14
6634                                <1> sysdma_13:
6635 00013D31 3B0D[B89F0100]       <1>       cmp   ecx, [dma_size]
6636 00013D37 0F87E5FEFFFF         <1>       ja    sysdma_err
6637                                <1> sysdma_14:
6638 00013D3D 89C6                 <1>       mov   esi, eax
6639 00013D3F 0335[B89F0100]       <1>       add   esi, [dma_size]
6640                                <1>
6641 00013D45 89D0                 <1>       mov   eax, edx
6642 00013D47 01C8                 <1>       add   eax, ecx
6643 00013D49 0F82D3FEFFFF         <1>       jc    sysdma_err ; 02/09/2017
6644                                <1>
6645 00013D4F 39F0                 <1>       cmp   eax, esi
6646 00013D51 0F87CBFEFFFF         <1>       ja    sysdma_err
6647                                <1>
6648 00013D57 8B3D[909B0100]       <1>       mov   edi, [audio_dma_buff]
6649 00013D5D 8B35[B49F0100]       <1>       mov   esi, [dma_addr]
6650                                <1>
6651 00013D63 09FF                 <1>       or    edi, edi
6652 00013D65 7424                 <1>       jz    short sysdma_16
6653                                <1>
6654 00013D67 803D[719B0100]01     <1>       cmp   byte [audio_device], 1
6655 00013D6E 7208                 <1>       jb    short sysdma_15
6656                                <1>
6657                                <1>       ; Sound Blaster 16
6658 00013D70 39FE                 <1>       cmp   esi, edi
6659 00013D72 0F84F4FEFFFF         <1>       je    sysdma_0_err ; permmission denied !
6660                                <1>
6661                                <1> sysdma_15:
6662 00013D78 C605[B39F0100]48     <1>       mov   byte [dma_mode], 48h ; single mode playback
6663                                <1>
6664 00013D7F F6C380               <1>       test  bl, 80h ; DMA mode - BL, bit 7, auto init -
6665 00013D82 7407                 <1>       jz    short sysdma_16
6666                                <1>       ; Auto initialized playback (write) mode
6667 00013D84 8005[B39F0100]10     <1>       add   byte [dma_mode], 10h ; = 58h
6668                                <1> sysdma_16:
6669 00013D8B 80E307               <1>       and   bl, 07h
6670 00013D8E 881D[B29F0100]       <1>       mov   [dma_channel], bl
6671 00013D94 8915[BC9F0100]       <1>       mov   [dma_start], edx
6672 00013D9A 890D[C09F0100]       <1>       mov   [dma_count], ecx
6673                                <1>
6674                                <1>       ; 28/08/2017
6675                                <1>       ;call  dma_init
6676                                <1>       ;jmp   sysret
6677 00013DA0 E94B010000           <1>       jmp   dma_init
6678                                <1>
6679                                <1> sysdma_5:
6680 00013DA5 80FF05               <1>       cmp   bh, 5
6681 00013DA8 7223                 <1>       jb    short sysdma_3
6682 00013DAA 0F87CE000000         <1>       ja    sysdma_6
6683                                <1>
6684                                <1>       ; Get the system's default dma buffer addr and size
6685 00013DB0 B8[00000200]         <1>       mov   eax, sb16_dma_buffer
6686 00013DB5 B900000100           <1>       mov   ecx, 65536 ; Buffer size in bytes
6687                                <1>
6688                                <1> sysdma_51:
6689                                <1>       ; 0 = there is not a dma buffer (in use or available)
6690 00013DBA A3[64030300]         <1>       mov   [u.r0], eax
6691                                <1>
6692 00013DBF 8B2D[60030300]       <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
6693 00013DC5 894D18               <1>       mov   [ebp+24], ecx ; return to user with ecx value
6694                                <1>
6695 00013DC8 E9FF99FFFF           <1>       jmp   sysret
6696                                <1>
6697                                <1> sysdma_3:
6698 00013DCD 80FF03               <1>       cmp   bh, 3
6699 00013DD0 7231                 <1>       jb    short sysdma_2
6700 00013DD2 776B                 <1>       ja    short sysdma_4
6701                                <1>
6702                                <1>       ; Get current dma count down value (remain bytes)
6703                                <1>       ; 28/08/2017
6704 00013DD4 0FB635[B29F0100]     <1>       movzx  esi, byte [dma_channel]
6705 00013DDB 0FB696[F0480100]     <1>       movzx  edx, byte [dma_flip+esi]
6706 00013DE2 EE                   <1>       out    dx, al              ; flip-flop clear
6707 00013DE3 8A96[D0480100]       <1>       mov    dl, [dma_cnt+esi] ; dma count register addr
6708 00013DE9 EC                   <1>       in     al, dx
6709 00013DEA 0FB6D8               <1>       movzx  ebx, al
6710 00013DED EC                   <1>       in     al, dx
6711 00013DEE 88C7                 <1>       mov     bh, al
6712                                <1>
6713 00013DF0 6683FE04             <1>       cmp    si, 4 ; channel number ?
6714 00013DF4 7202                 <1>       jb     short sysdma_31 ; 8 bit dma channel
6715                                <1>
6716 00013DF6 D1E3                 <1>       shl    ebx, 1 ; word count to byte count
6717                                <1>
6718                                <1> sysdma_31:
6719 00013DF8 891D[64030300]       <1>       mov    [u.r0], ebx
6720                                <1>
6721 00013DFE E9C999FFFF           <1>       jmp    sysret
6722                                <1>
6723                                <1> sysdma_2:
6724                                <1>       ; Get current dma buffer offset (& page)
6725                                <1>       ; 28/08/2017
6726 00013E03 0FB635[B29F0100]     <1>       movzx  esi, byte [dma_channel]
6727 00013E0A 0FB696[F0480100]     <1>       movzx  edx, byte [dma_flip+esi]
6728 00013E11 EE                   <1>       out    dx, al              ; flip-flop clear
6729 00013E12 8A96[C8480100]       <1>       mov    dl, [dma_adr+esi]
6730 00013E18 EC                   <1>       in     al, dx              ; get dma position
6731 00013E19 0FB6D8               <1>       movzx  ebx, al
6732 00013E1C EC                   <1>       in     al, dx
6733 00013E1D 88C7                 <1>       mov    bh, al
6734                                <1>
6735 00013E1F 6683FE04             <1>       cmp    si, 4 ; channel number ?
```

```
6736 00013E23 7202                <1>       jb    short sysdma_21 ; 8 bit dma channel
6737                              <1>
6738 00013E25 D1E3                <1>       shl   ebx, 1 ; word offset to byte offset
6739                              <1>
6740                              <1> sysdma_21:
6741 00013E27 891D[64030300]      <1>       mov   [u.r0], ebx
6742                              <1>
6743 00013E2D 8A96[D8480100]      <1>       mov   dl, [dma_page+esi]
6744 00013E33 EC                  <1>       in    al, dx              ; get dma page
6745                              <1>
6746                              <1>       ;add  [u.ro+2], al
6747 00013E34 0805[66030300]      <1>       or    [u.r0+2], al
6748                              <1>
6749 00013E3A E98D99FFFF          <1>       jmp   sysret
6750                              <1>
6751                              <1> sysdma_4:
6752                              <1>       ; Get current DMA channel number
6753                              <1>       ; 28/08/2017
6754 00013E3F 8A25[B19F0100]      <1>       mov   ah, [dma_user]
6755 00013E45 20E4                <1>       and   ah, ah
6756 00013E47 750F                <1>       jnz   short sysdma_42
6757                              <1>
6758                              <1> sysdma_41:
6759                              <1>       ; Not a valid dma channel (in use)
6760 00013E49 C705[64030300]FFFF- <1>       mov   dword [u.r0], -1 ; 0FFFFFFFFh
6760 00013E51 FFFF                <1>
6761 00013E53 E97499FFFF          <1>       jmp   sysret
6762                              <1>
6763                              <1> sysdma_42:
6764 00013E58 8B35[B49F0100]      <1>       mov   esi, [dma_addr]
6765 00013E5E 21F6                <1>       and   esi, esi
6766 00013E60 74E7                <1>       jz    short sysdma_41
6767                              <1>
6768 00013E62 46                  <1>       inc   esi ; -1 -> 0
6769 00013E63 74E4                <1>       jz    short sysdma_41
6770                              <1>
6771 00013E65 A0[B29F0100]        <1>       mov   al, [dma_channel]
6772                              <1>
6773 00013E6A 3A25[B3030300]      <1>       cmp   ah, [u.uno]
6774 00013E70 7502                <1>       jne   short sysdma_43
6775                              <1>
6776 00013E72 30E4                <1>       xor   ah, ah ; DMA channel in use by current user
6777                              <1>
6778                              <1> sysdma_43:
6779 00013E74 A3[64030300]        <1>       mov   [u.r0], eax ; AL = dma channel number
6780                              <1>                         ; AH > 0 if the the channel
6781                              <1>                         ; in use by another user/process
6782 00013E79 E94E99FFFF          <1>       jmp   sysret
6783                              <1>
6784                              <1> sysdma_6:
6785 00013E7E 80FF06              <1>       cmp   bh, 6
6786 00013E81 7710                <1>       ja    short sysdma_7
6787                              <1>
6788                              <1>       ; 28/08/2017
6789                              <1>       ; Get current DMA buffer addr and size
6790 00013E83 A1[B49F0100]        <1>       mov   eax, [dma_addr] ; dma buffer address
6791 00013E88 8B0D[B89F0100]      <1>       mov   ecx, [dma_size] ; dma buffer size (in bytes)
6792                              <1>
6793 00013E8E E927FFFFFF          <1>       jmp   sysdma_51
6794                              <1>
6795                              <1> sysdma_7:
6796                              <1>       ; DMA service STOP
6797 00013E93 A0[B3030300]        <1>       mov   al, [u.uno]
6798 00013E98 3A05[B19F0100]      <1>       cmp   al, [dma_user]
6799 00013E9E 751D                <1>       jne   short sysdma_72
6800                              <1>
6801 00013EA0 28C0                <1>       sub   al, al ; 0
6802                              <1>
6803 00013EA2 A2[B19F0100]        <1>       mov   [dma_user], al ; clear user
6804                              <1>
6805 00013EA7 8605[B39F0100]      <1>       xchg  al, [dma_mode]
6806 00013EAD 20C0                <1>       and   al, al
6807                              <1>       ;jz   short sysdma_err
6808 00013EAF 7527                <1>       jnz   short sysdma_73
6809                              <1>
6810                              <1> sysdma_71:
6811 00013EB1 31C0                <1>       xor   eax, eax
6812 00013EB3 A3[64030300]        <1>       mov   [u.r0], eax; 0
6813 00013EB8 E90F99FFFF          <1>       jmp   sysret
6814                              <1>
6815                              <1> sysdma_72:
6816                              <1>       ; 28/08/2017
6817 00013EBD 803D[B19F0100]00    <1>       cmp   byte [dma_user], 0
6818 00013EC4 76EB                <1>       jna   short sysdma_71 ; Nothing to do !
6819                              <1>
6820 00013EC6 833D[B49F0100]00    <1>       cmp   dword [dma_addr], 0
6821 00013ECD 0F8799FDFFFF        <1>       ja    sysdma_0_err
6822                              <1>
6823 00013ED3 A2[B19F0100]        <1>       mov   [dma_user], al ; reset to current user
6824                              <1>
6825                              <1> sysdma_73:
6826                              <1>       ; 28/08/2017
6827 00013ED8 0FB635[B29F0100]    <1>       movzx esi, byte [dma_channel]
6828 00013EDF 0FB696[E0480100]    <1>       movzx edx, byte [dma_mask+esi]
6829 00013EE6 A0[B29F0100]        <1>       mov   al, [dma_channel]
6830 00013EEB 0C04                <1>       or    al, 4
6831 00013EED EE                  <1>       out   dx, al
6832                              <1>
6833 00013EEE EBC1                <1>       jmp   short sysdma_71
6834                              <1>
6835                              <1> dma_init:
6836                              <1>       ; 28/08/2017
6837                              <1>       ; 20/08/2017
6838                              <1>       ; DMA initialization
6839                              <1>       ; 14/08/2017
```

```
6840                            <1>       ; 03/08/2017, 06/08/2017, 08/08/2017
6841                            <1>       ; 02/07/2017, 13/07/2017, 16/07/2017, 30/07/2017
6842                            <1>       ; (Derived from 'DMA_INIT' procedure in SB16MOD.ASM)
6843                            <1>       ; Modified for TRDOS 386 DMA buffer allocation & initialization !
6844                            <1>
6845 00013EF0 8B1D[BC9F0100]  <1>       mov    ebx, [dma_start]
6846 00013EF6 8B0D[C09F0100]  <1>       mov    ecx, [dma_count]
6847                            <1>
6848 00013EFC 0FB635[B29F0100] <1>       movzx  esi, byte [dma_channel]
6849                            <1>
6850 00013F03 6683FE04        <1>       cmp    si, 4
6851 00013F07 7205            <1>       jb     short gdmi1
6852                            <1>       ; 08/08/2017
6853 00013F09 66D1E9          <1>       shr    cx, 1 ; word count
6854 00013F0C D1EB            <1>       shr    ebx, 1 ; convert byte offset to word offset
6855                            <1> gdmi1:
6856                            <1>       ;mov   [dma_poff], bx ; 08/08/2017
6857 00013F0E 6649            <1>       dec    cx                  ; dma size = block size - 1
6858                            <1>
6859 00013F10 0FB696[E0480100] <1>       movzx  edx, byte [dma_mask+esi] ; 30/07/2017
6860 00013F17 A0[B29F0100]    <1>       mov    al, [dma_channel]
6861 00013F1C 0C04            <1>       or     al, 4
6862 00013F1E EE              <1>       out    dx, al              ; dma channel mask
6863                            <1>
6864 00013F1F 30C0            <1>       xor    al, al ; 0 ; any value ! 08/08/2017
6865 00013F21 8A96[F0480100]  <1>       mov    dl, [dma_flip+esi]
6866 00013F27 EE              <1>       out    dx, al              ; flip-flop clear
6867                            <1>
6868 00013F28 8A96[E8480100]  <1>       mov    dl, [dma_mod+esi]
6869 00013F2E A0[B29F0100]    <1>       mov    al, [dma_channel]  ; 13/07/2017
6870 00013F33 2403            <1>       and    al, 3
6871                            <1>       ; 08/08/2017
6872 00013F35 0A05[B39F0100]  <1>       or     al, [dma_mode] ; 58h    ; dma mode for SB16
6873 00013F3B EE              <1>       out    dx, al
6874                            <1>
6875 00013F3C 8A96[C8480100]  <1>       mov    dl, [dma_adr+esi]
6876 00013F42 88D8            <1>       mov    al, bl
6877 00013F44 EE              <1>       out    dx, al              ; offset low
6878                            <1>
6879 00013F45 88F8            <1>       mov    al, bh
6880 00013F47 EE              <1>       out    dx, al              ; offset high
6881                            <1>
6882 00013F48 8A96[D0480100]  <1>       mov    dl, [dma_cnt+esi]
6883 00013F4E 88C8            <1>       mov    al, cl
6884 00013F50 EE              <1>       out    dx, al              ; size low
6885                            <1>
6886 00013F51 88E8            <1>       mov    al, ch
6887 00013F53 EE              <1>       out    dx, al              ; size high
6888                            <1>
6889 00013F54 8A96[D8480100]  <1>       mov    dl, [dma_page+esi]
6890                            <1>       ; 14/08/2017
6891 00013F5A 6683FE04        <1>       cmp    si, 4
6892 00013F5E 7305            <1>       jnb    short gdmi2
6893 00013F60 C1EB10          <1>       shr    ebx, 16
6894 00013F63 EB06            <1>       jmp    short gdmi3
6895                            <1> gdmi2:
6896                            <1>       ; 09/08/2017
6897 00013F65 C1EB0F          <1>       shr    ebx, 15       ; complete 16 bit shift
6898 00013F68 80E3FE          <1>       and    bl, 0FEh ; clear bit 0 (not necessary)
6899                            <1> gdmi3:
6900 00013F6B 88D8            <1>       mov    al, bl
6901 00013F6D EE              <1>       out    dx, al              ; page
6902                            <1>
6903 00013F6E 8A96[E0480100]  <1>       mov    dl, [dma_mask+esi]
6904 00013F74 A0[B29F0100]    <1>       mov    al, [dma_channel]  ; 13/07/2017
6905 00013F79 2403            <1>       and    al, 3
6906 00013F7B EE              <1>       out    dx, al              ; dma channel unmask
6907                            <1>
6908                            <1>       ;retn
6909                            <1>       ; 28/08/2017
6910 00013F7C E94B98FFFF      <1>       jmp    sysret
6911                            <1>
6912                            <1> otty:
6913                            <1> sret:
6914                            <1> ocvt:
6915                            <1> ctty:
6916                            <1> cret:
6917                            <1> ccvt:
6918                            <1> rtty:
6919                            <1> wtty:
6920                            <1> rmem:
6921                            <1> wmem:
6922                            <1> rfd:
6923                            <1> rhd:
6924                            <1> wfd:
6925                            <1> whd:
6926                            <1> rlpt:
6927                            <1> wlpt:
6928                            <1> rcvt:
6929                            <1> xmtt:
6930 00013F81 C3                <1>       retn
3092                                %include 'trdosk9.s' ; 04/01/2016
3093                            <1> ; ****************************************************************************
3094                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.3) - INITIALIZED DATA : trdosk9.s
3095                            <1> ; ----------------------------------------------------------------------------
3096                            <1> ; Last Update: 12/04/2021
3097                            <1> ; ----------------------------------------------------------------------------
3098                            <1> ; Beginning: 04/01/2016
3099                            <1> ; ----------------------------------------------------------------------------
3100                            <1> ; ----------------------------------------------------------------------------
3101                            <1> ; Assembler: NASM version 2.15 (trdos386.s)
3102                            <1> ; ----------------------------------------------------------------------------
3103                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3104                            <1> ; TRDOS2.ASM (09/11/2011)
3105                            <1> ; ****************************************************************************
```

```
3106                                    <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
3107                                    <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
3108                                    <1> ; CMD_INTR.ASM [29/01/2005] Last Update: 09/11/2011
3109                                    <1> ; FILE.ASM [29/10/2009] Last Update: 09/10/2011
3110                                    <1>
3111                                    <1> ; 12/02/2016
3112                                    <1> Last_DOS_DiskNo:
3113 00013F82 01                       <1>             db 1 ; A: = 0 & B: = 1
3114                                    <1>
3115                                    <1> Restore_CDIR:
3116 00013F83 FF                       <1>             db 0FFh ; Initial value -> any number except 0
3117                                    <1>
3118                                    <1> msg_CRLF_temp:
3119 00013F84 070D0A00                 <1>             db 07h, 0Dh, 0Ah, 0
3120                                    <1>
3121                                    <1> Magic_Bytes:
3122 00013F88 04                       <1>             db 4
3123 00013F89 01                       <1>             db 1
3124                                    <1> mainprog_Version:
3125 00013F8A 07                       <1>             db 7
3126 00013F8B 5B5452444F535D204D-      <1>             db "[TRDOS] Main Program v2.0.120421"
3126 00013F94 61696E2050726F6772-      <1>
3126 00013F9D 616D2076322E302E31-      <1>
3126 00013FA6 3230343231               <1>
3127 00013FAB 0D0A                     <1>             db 0Dh, 0Ah
3128 00013FAD 286329204572646F67-      <1>             db "(c) Erdogan Tan 2005-2021"
3128 00013FB6 616E2054616E203230-      <1>
3128 00013FBF 30352D32303231           <1>
3129 00013FC6 0D0A00                   <1>             db 0Dh, 0Ah, 0
3130                                    <1>
3131                                    <1> MainProgCfgFile: ; 14/04/2016
3132 00013FC9 4D41494E50524F472E-      <1>             db "MAINPROG.CFG", 0
3132 00013FD2 43464700                 <1>
3133                                    <1>
3134                                    <1> TRDOSPromptLabel:
3135 00013FD6 5452444F53               <1>             db "TRDOS"
3136 00013FDB 00                       <1>             db 0
3137 00013FDC 00<rep 5h>               <1>                times 5 db 0
3138 00013FE1 00                       <1>             db 0
3139                                    <1>
3140                                    <1> ; INTERNAL COMMANDS
3141                                    <1> Command_List:
3142 00013FE2 44495200                 <1> Cmd_Dir:    db "DIR", 0
3143 00013FE6 434400                   <1> Cmd_Cd:        db "CD", 0
3144 00013FE9 433A00                   <1> Cmd_Drive:  db "C:", 0
3145 00013FEC 56455200                 <1> Cmd_Ver:    db "VER", 0
3146 00013FF0 4558495400               <1> Cmd_Exit:   db "EXIT", 0
3147 00013FF5 50524F4D505400           <1> Cmd_Prompt: db "PROMPT", 0
3148 00013FFC 564F4C554D4500           <1> Cmd_Volume: db "VOLUME", 0
3149 00014003 4C4F4E474E414D4500       <1> Cmd_LongName:     db "LONGNAME", 0
3150 0001400C 4441544500               <1> Cmd_Date:   db "DATE", 0
3151 00014011 54494D4500               <1> Cmd_Time:   db "TIME", 0
3152 00014016 52554E00                 <1> Cmd_Run:    db "RUN", 0
3153 0001401A 53455400                 <1> Cmd_Set:    db "SET", 0
3154 0001401E 434C5300                 <1> Cmd_Cls:    db "CLS", 0
3155 00014022 53484F5700               <1> Cmd_Show:   db "SHOW", 0
3156 00014027 44454C00                 <1> Cmd_Del:    db "DEL", 0
3157 0001402B 41545452494200           <1> Cmd_Attrib: db "ATTRIB", 0
3158 00014032 52454E414D4500           <1> Cmd_Rename: db "RENAME", 0
3159 00014039 524D44495200             <1> Cmd_Rmdir:  db "RMDIR", 0
3160 0001403F 4D4B44495200             <1> Cmd_Mkdir:  db "MKDIR", 0
3161 00014045 434F505900               <1> Cmd_Copy:   db "COPY", 0
3162 0001404A 4D4F564500               <1> Cmd_Move:   db "MOVE", 0
3163 0001404F 5041544800               <1> Cmd_Path:   db "PATH", 0
3164 00014054 4D454D00                 <1> Cmd_Mem:    db "MEM", 0
3165 00014058 00                       <1>             db 0
3166 00014059 46494E4400               <1> Cmd_Find:   db "FIND", 0
3167 0001405E 4543484F00               <1> Cmd_Echo:   db "ECHO", 0
3168 00014063 2A00                     <1> Cmd_Remark: db "*", 0
3169 00014065 3F00                     <1> Cmd_Help:   db "?", 0
3170 00014067 44455649434500           <1> Cmd_Device: db "DEVICE", 0
3171 0001406E 4445564C49535400         <1> Cmd_DevList: db "DEVLIST", 0
3172 00014076 434844495200             <1> Cmd_Chdir:  db "CHDIR", 0
3173 0001407C 4245455000               <1> Cmd_Beep:   db "BEEP", 0
3174                                    <1>
3175 00014081 00                       <1>             db 0
3176                                    <1>
3177                                    <1> ; 15/02/2016 (FILE.ASM, 09/10/2011)
3178                                    <1> invalid_fname_chars:
3179 00014082 222728292A2B2C2F         <1>             db 22h, 27h, 28h, 29h, 2Ah, 2Bh, 2Ch, 2Fh
3180 0001408A 3A3B3C3D3E3F40           <1>             db 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh, 40h
3181 00014091 5B5C5D5E60               <1>             db 5Bh, 5Ch, 5Dh, 5Eh, 60h
3182                                    <1> sizeInvFnChars  equ ($ - invalid_fname_chars)
3183                                    <1> ;
3184                                    <1>
3185                                    <1> Msg_Enter_Date:
3186 00014096 456E746572206E6577-      <1>                 db 'Enter new date (dd-mm-yy): '
3186 0001409F 206461746520286464-      <1>
3186 000140A8 2D6D6D2D7979293A20       <1>
3187 000140B1 00                       <1>                 db 0
3188                                    <1> Msg_Show_Date:
3189 000140B2 43757272656E742064-      <1>                 db   'Current date is '
3189 000140BB 61746520697320           <1>
3190 000140C2 30                       <1> Day:            db   '0'
3191 000140C3 30                       <1>             db   '0'
3192 000140C4 2F                       <1>                 db   '/'
3193 000140C5 30                       <1> Month:          db   '0'
3194 000140C6 30                       <1>             db   '0'
3195 000140C7 2F                       <1>                 db   '/'
3196 000140C8 30                       <1> Century:        db   '0'
3197 000140C9 30                       <1>                 db   '0'
3198 000140CA 30                       <1> Year:           db   '0'
3199 000140CB 30                       <1>             db   '0'
3200 000140CC 0D0A00                   <1>                 db   0Dh, 0Ah, 0
3201                                    <1>
```

```
3202                                <1> Msg_Enter_Time:
3203 000140CF 456E746572206E6577-   <1>             db 'Enter new time: '
3203 000140D8 2074696D653A20        <1>
3204 000140DF 00                    <1>             db 0
3205                                <1> Msg_Show_Time:
3206 000140E0 43757272656E742074-   <1>             db   'Current time is '
3206 000140E9 696D6520697320        <1>
3207 000140F0 30                    <1> Hour:           db   '0'
3208 000140F1 30                    <1>             db   '0'
3209 000140F2 3A                    <1>             db   ':'
3210 000140F3 30                    <1> Minute:         db   '0'
3211 000140F4 30                    <1>             db   '0'
3212 000140F5 3A                    <1>             db   ':'
3213 000140F6 30                    <1> Second:         db   '0'
3214 000140F7 30                    <1>             db   '0'
3215 000140F8 0D0A00                <1>             db   0Dh, 0Ah, 0
3216                                <1>
3217                                <1> ;VolSize_Unit1:   dd 0
3218                                <1> ;VolSize_Unit2:   dd 0
3219                                <1>
3220                                <1> VolSize_KiloBytes:
3221 000140FB 206B696C6F62797465-   <1>             db " kilobytes", 0Dh, 0Ah, 0
3221 00014104 730D0A00              <1>
3222                                <1> VolSize_Bytes:
3223 00014108 2062797465730D0A00    <1>             db " bytes", 0Dh, 0Ah, 0
3224                                <1> Volume_in_drive:
3225 00014111 0D0A                  <1>             db 0Dh, 0Ah
3226                                <1> Vol_FS_Name:
3227 00014113 54522046533120        <1>             db "TR FS1 "
3228 0001411A 566F6C756D6520696E-   <1>             db "Volume in drive "
3228 00014123 20647269766520        <1>
3229 0001412A 30                    <1> Vol_Drv_Name:   db 30h
3230 0001412B 3A                    <1>             db ":"
3231 0001412C 20697320              <1>             db " is "
3232 00014130 0D0A00                <1>             db 0Dh, 0Ah, 0
3233                                <1> Dir_Drive_Str:
3234 00014133 54522D444F53204472-   <1>                 db "TR-DOS Drive "
3234 0001413C 69766520              <1>
3235                                <1> Dir_Drive_Name:
3236 00014140 303A                  <1>                 db "0:"
3237 00014142 0D0A                  <1>                 db  0Dh, 0Ah
3238                                <1> Vol_Str_Header:
3239 00014144 566F6C756D65204E61-   <1>                 db "Volume Name: "
3239 0001414D 6D653A20              <1>
3240                                <1> Vol_Name:
3241 00014151 00<rep 40h>           <1>             times 64 db 0
3242 00014191 00                    <1>             db 0
3243                                <1> Vol_Serial_Header:
3244 00014192 0D0A                  <1>             db 0Dh, 0Ah
3245 00014194 566F6C756D65205365-   <1>             db "Volume Serial No: "
3245 0001419D 7269616C204E6F3A20    <1>
3246                                <1> Vol_Serial1:
3247 000141A6 30303030              <1>             db "0000"
3248 000141AA 2D                    <1>             db "-"
3249                                <1> Vol_Serial2:
3250 000141AB 30303030              <1>             db "0000"
3251 000141AF 0D0A00                <1>             db 0Dh, 0Ah, 0
3252                                <1>
3253                                <1> ;Vol_Tot_Sec_Str_Start:
3254                                <1> ;           dd 0
3255                                <1> Vol_Total_Sector_Header:
3256 000141B2 0D0A                  <1>             db 0Dh, 0Ah
3257 000141B4 566F6C756D65205369-   <1>             db "Volume Size : ", 0
3257 000141BD 7A65203A2000          <1>
3258                                <1> ;Vol_Tot_Sec_Str:
3259                                <1> ;           db "0000000000"
3260                                <1> ;Vol_Tot_Sec_Str_End:
3261                                <1> ;           db 0
3262                                <1> ;Vol_Free_Sectors_Str_Start:
3263                                <1> ;           dd 0
3264                                <1> Vol_Free_Sectors_Header:
3265 000141C3 467265652053706163-   <1>             db "Free Space  : ", 0
3265 000141CC 6520203A2000          <1>
3266                                <1> ;Vol_Free_Sectors_Str:
3267                                <1> ;           db "0000000000"
3268                                <1> ;Vol_Free_Sectors_Str_End:
3269                                <1> ;           db 0
3270                                <1>
3271                                <1> Dir_Str_Header:
3272 000141D2 4469726563746F7279-   <1>                 db "Directory: "
3272 000141DB 3A20                  <1>
3273 000141DD 2F                    <1> Dir_Str_Root:   db "/"
3274 000141DE 00<rep 40h>           <1> Dir_Str:        times 64 db 0
3275 0001421E 00000000              <1>                 dd 0
3276 00014222 00                    <1>                 db 0
3277                                <1>
3278                                <1> Msg_Bad_Command:
3279 00014223 42616420636F6D6D61-   <1>                 db "Bad command or file name!"
3279 0001422C 6E64206F722066696C-   <1>
3279 00014235 65206E616D6521        <1>
3280 0001423C 0D0A00                <1>                 db 0Dh, 0Ah, 0
3281                                <1>
3282                                <1> msgl_drv_not_ready:
3283 0001423F 070D0A                <1>             db 07h, 0Dh, 0Ah
3284                                <1>
3285                                <1> ; CMD_INTR.ASM - 09/11/2011 - Messages
3286                                <1>
3287                                <1> Msg_Not_Ready_Read_Err:
3288 00014242 4472697665206E6F74-   <1>                 db "Drive not ready or read error!"
3288 0001424B 207265616479206F72-   <1>
3288 00014254 207265616420657272-   <1>
3288 0001425D 6F7221                <1>
3289 00014260 0D0A00                <1>                 db 0Dh, 0Ah, 0
3290                                <1>
3291                                <1> Msg_Not_Ready_Write_Err:
```

```
3292 00014263 4472697665206E6F74- <1>                db "Drive not ready or write error!"
3292 0001426C 207265616479206F72- <1>
3292 00014275 207772697465206572- <1>
3292 0001427E 726F7221            <1>
3293 00014282 0D0A00              <1>                db 0Dh, 0Ah, 0
3294                              <1>
3295                              <1> Msg_Dir_Not_Found:
3296 00014285 4469726563746F7279- <1>                db "Directory not found!"
3296 0001428E 206E6F7420666F756E- <1>
3296 00014297 6421                <1>
3297 00014299 0D0A00              <1>                db 0Dh, 0Ah, 0
3298                              <1>
3299                              <1> Msg_File_Not_Found:
3300 0001429C 46696C65206E6F7420- <1>                db "File not found!"
3300 000142A5 666F756E6421        <1>
3301 000142AB 0D0A00              <1>                db 0Dh, 0Ah, 0
3302                              <1>
3303                              <1> Msg_File_Directory_Not_Found:
3304 000142AE 46696C65206F722064- <1>                db "File or directory not found!"
3304 000142B7 69726563746F727920- <1>
3304 000142C0 6E6F7420666F756E64- <1>
3304 000142C9 21                  <1>
3305 000142CA 0D0A00              <1>                db 0Dh, 0Ah, 0
3306                              <1>
3307                              <1> Msg_LongName_Not_Found:
3308 000142CD 4C6F6E67206E616D65- <1>                db "Long name not found!"
3308 000142D6 206E6F7420666F756E- <1>
3308 000142DF 6421                <1>
3309 000142E1 0D0A00              <1>                db 0Dh, 0Ah, 0
3310                              <1>
3311                              <1> beep_Insufficient_Memory: ; 20/02/2017
3312 000142E4 0D0A                <1>                db 0Dh, 0Ah
3313 000142E6 07                  <1>                db 07h
3314                              <1> Msg_Insufficient_Memory:
3315 000142E7 496E73756666696369- <1>                db "Insufficient memory!"
3315 000142F0 656E74206D656D6F72- <1>
3315 000142F9 7921                <1>
3316 000142FB 0D0A00              <1>                db 0Dh, 0Ah, 0
3317                              <1>
3318                              <1> Msg_Error_Code:
3319 000142FE 436F6D6D616E642066- <1>                db 'Command failed! Error code : '
3319 00014307 61696C656421204572- <1>
3319 00014310 726F7220636F646520- <1>
3319 00014319 3A20                <1>
3320 0001431B 303068              <1> error_code_hex: db '00h'
3321 0001431E 0A0A00              <1>                db 0Ah, 0Ah, 0
3322                              <1>
3323 00014321 90                  <1> align 2
3324                              <1>
3325                              <1> ; 10/02/2016
3326                              <1> ; DIR.ASM - 09/10/2011
3327                              <1>
3328 00014322 3C4449523E20202020- <1> Type_Dir:       db '<DIR>     ' ; 10 bytes
3328 0001432B 20                  <1>
3329                              <1>
3330                              <1> File_Name:
3331 0001432C 20<rep Ch>          <1>                times 12 db 20h
3332 00014338 20                  <1>              db 20h
3333                              <1> Dir_Or_FileSize:
3334 00014339 20<rep Ah>          <1>                times 10 db 20h
3335 00014343 20                  <1>              db 20h
3336                              <1> File_Attribute:
3337 00014344 20202020            <1>              dd 20202020h
3338 00014348 20                  <1>              db 20h
3339                              <1> File_Day:
3340 00014349 3030                <1>                 db '0','0'
3341 0001434B 2F                  <1>              db '/'
3342                              <1> File_Month:
3343 0001434C 3030                <1>                 db '0','0'
3344 0001434E 2F                  <1>              db '/'
3345                              <1> File_Year:
3346 0001434F 30303030            <1>                 db '0','0','0','0'
3347 00014353 20                  <1>              db 20h
3348                              <1> File_Hour:
3349 00014354 3030                <1>                 db '0','0'
3350 00014356 3A                  <1>              db ':'
3351                              <1> File_Minute:
3352 00014357 3030                <1>                 db '0','0'
3353 00014359 00                  <1>              db 0
3354                              <1>
3355                              <1> Decimal_File_Count_Header:
3356 0001435A 0D0A                <1>              db 0Dh, 0Ah
3357                              <1> Decimal_File_Count:
3358 0001435C 00<rep 6h>          <1>              times 6 db 0
3359                              <1>
3360 00014362 2066696C6528732920- <1> str_files:  db " file(s) & "
3360 0001436B 2620                <1>
3361                              <1> Decimal_Dir_Count:
3362 0001436D 00<rep 6h>          <1>              times 6 db 0
3363                              <1> str_dirs:
3364 00014373 206469726563746F72- <1>              db " directory(s) "
3364 0001437C 7928732920          <1>
3365 00014381 0D0A00              <1>              db 0Dh, 0Ah, 0
3366                              <1>
3367 00014384 206279746528732920- <1> str_bytes:  db " byte(s) in file(s)"
3367 0001438D 696E2066696C652873- <1>
3367 00014396 29                  <1>
3368 00014397 0D0A00              <1>              db 0Dh, 0Ah, 0
3369                              <1>
3370                              <1> ; CMD_INTR.ASM - 09/11/2011
3371                              <1> ; 07/10/2010
3372                              <1> Msg_invalid_name_chars:
3373 0001439A 496E76616C69642066- <1>                db "Invalid file or directory name characters!"
3373 000143A3 696C65206F72206469- <1>
3373 000143AC 726563746F7279206E- <1>
```

```
3373 000143B5 616D65206368617261- <1>
3373 000143BE 637465727321        <1>
3374 000143C4 0D0A00              <1>             db 0Dh, 0Ah, 0
3375                              <1> ; 21/02/2016
3376 000143C7 46696C65206F722064- <1> Msg_Name_Exists: db "File or directory name exists!"
3376 000143D0 69726563746F727920- <1>
3376 000143D9 6E616D652065786973- <1>
3376 000143E2 747321              <1>
3377 000143E5 0D0A00              <1>                 db 0Dh, 0Ah, 0
3378                              <1> Msg_DoYouWantMkdir:
3379 000143E8 446F20796F75207761- <1>                 db "Do you want to make directory ", 0
3379 000143F1 6E7420746F206D616B- <1>
3379 000143FA 65206469726563746F- <1>
3379 00014403 72792000            <1>
3380 00014407 2028592F4E29203F20- <1> Msg_YesNo:      db " (Y/N) ? ", 0
3380 00014410 00                  <1>
3381 00014411 000D0A00            <1> Y_N_nextline:      db 0, 0Dh, 0Ah, 0
3382 00014415 4F4B2E0D0A00        <1> Msg_OK:            db "OK.", 0Dh, 0Ah, 0
3383                              <1>
3384                              <1> ; 27/02/2016
3385                              <1> Msg_DoYouWantRmDir:
3386 0001441B 446F20796F75207761- <1>                 db "Do you want to delete directory ", 0
3386 00014424 6E7420746F2064656C- <1>
3386 0001442D 657465206469726563- <1>
3386 00014436 746F72792000        <1>
3387                              <1> Msg_Dir_Not_Empty:
3388 0001443C 4469726563746F7279- <1>                 db "Directory not empty!"
3388 00014445 206E6F7420656D7074- <1>
3388 0001444E 7921                <1>
3389 00014450 0D0A00              <1>                 db 0Dh, 0Ah, 0
3390                              <1>
3391                              <1> Msg_DoYouWantDelete:
3392 00014453 446F20796F75207761- <1>                 db "Do you want to delete file ",0
3392 0001445C 6E7420746F2064656C- <1>
3392 00014465 6574652066696C6520- <1>
3392 0001446E 00                  <1>
3393                              <1>
3394 0001446F 44656C657465642E2E- <1> Msg_Deleted:    db "Deleted...", 0Dh, 0Ah, 0
3394 00014478 2E0D0A00            <1>
3395                              <1>
3396                              <1> Msg_Permission_Denied:
3397 0001447C 07                  <1>                 db 7
3398 0001447D 5065726D697373696F- <1>                 db "Permission denied!", 0Dh, 0Ah, 0
3398 00014486 6E2064656E69656421- <1>
3398 0001448F 0D0A00              <1>
3399                              <1>
3400                              <1> ; 04/03/2016
3401 00014492 4E657720            <1> Msg_New:        db "New "
3402 00014496 00                  <1>                 db 0
3403                              <1> Str_Attributes:
3404 00014497 417474726962757465- <1>                 db "Attributes : "
3404 000144A0 73203A20            <1>
3405 000144A4 4E4F524D414C        <1> Attr_Chars:     db "NORMAL"
3406 000144AA 00                  <1>                 db 0
3407                              <1>
3408                              <1> ; 06/03/2016
3409                              <1> ; CMD_INTR.ASM - 16/11/2010
3410                              <1> Msg_DoYouWantRename:
3411 000144AB 446F20796F75207761- <1>                 db "Do you want to rename ", 0
3411 000144B4 6E7420746F2072656E- <1>
3411 000144BD 616D652000          <1>
3412 000144C2 66696C652000        <1> Rename_File:    db "file ", 0
3413 000144C8 6469726563746F7279- <1> Rename_Directory: db "directory ", 0
3413 000144D1 2000                <1>
3414 000144D3 00<rep Dh>          <1> Rename_OldName: times 13 db 0
3415 000144E0 20617320            <1> Msg_File_rename_as: db " as "
3416 000144E4 00<rep Dh>          <1> Rename_NewName: times 13 db 0
3417                              <1>
3418                              <1> ; 08/03/2016
3419                              <1> ; CMD_INTR.ASM - 01/08/2010 - 23/04/2011
3420                              <1> msg_not_same_drv:
3421 000144F1 4E6F742073616D6520- <1>                 db "Not same drive!"
3421 000144FA 647269766521        <1>
3422 00014500 0D0A00              <1>                 db 0Dh, 0Ah, 0
3423                              <1>
3424                              <1> Msg_DoYouWantMoveFile:
3425 00014503 446F20796F75207761- <1>                 db "Do you want to move file", 0
3425 0001450C 6E7420746F206D6F76- <1>
3425 00014515 652066696C6500      <1>
3426                              <1>
3427                              <1> msg_insufficient_disk_space:
3428 0001451C 496E73756666696369- <1>                 db "Insufficient disk space!"
3428 00014525 656E742064697B20- <1>
3428 0001452E 737061636521        <1>
3429 00014534 0D0A00              <1>                 db 0Dh, 0Ah, 0
3430                              <1>
3431                              <1> ; 01/08/2010
3432                              <1> msg_source_file:
3433 00014537 0D0A536F7572636520- <1>                 db 0Dh, 0Ah, "Source file name      :   "
3433 00014540 66696C65206E616D65- <1>
3433 00014549 2020202020203A2020- <1>
3433 00014552 20                  <1>
3434                              <1> msg_source_file_drv:
3435 00014553 203A00              <1>                 db " :", 0
3436                              <1> msg_destination_file:
3437 00014556 0D0A44657374696E61- <1>                 db 0Dh, 0Ah, "Destination file name :   "
3437 0001455F 74696F6E2066696C65- <1>
3437 00014568 206E616D65203A2020- <1>
3437 00014571 20                  <1>
3438                              <1> msg_destination_file_drv:
3439 00014572 203A00              <1>                 db " :", 0
3440                              <1> msg_copy_nextline:
3441 00014575 0D0A00              <1>                 db 0Dh, 0Ah, 0
3442                              <1>
3443                              <1> ; 15/03/2016
```

```
3444                               <1> ; CMD_INTR.ASM
3445                               <1>
3446                               <1> Msg_DoYouWantOverWriteFile:
3447 00014578 446F20796F75207761- <1>                 db "Do you want to overwrite file ",0
3447 00014581 6E7420746F206F7665- <1>
3447 0001458A 727772697465206669- <1>
3447 00014593 6C652000            <1>
3448                               <1>
3449                               <1> Msg_DoYouWantCopyFile:
3450 00014597 446F20796F75207761- <1>                 db "Do you want to copy file",0
3450 000145A0 6E7420746F20636F70- <1>
3450 000145A9 792066696C6500      <1>
3451                               <1>
3452                               <1> Msg_read_file_error_before_EOF:
3453 000145B0 46696C652072656164- <1>             db "File reading error! (before EOF)"
3453 000145B9 696E67206572726F72- <1>
3453 000145C2 2120286265666F7265- <1>
3453 000145CB 20454F4629          <1>
3454 000145D0 0A0A00              <1>             db 0Ah, 0Ah, 0
3455                               <1>
3456                               <1> ; 18/03/2016
3457                               <1> ; TRDOS 386 (v2.0) mainprog copy procedure
3458                               <1> msg_reading:
3459 000145D3 52656164696E672E2E- <1>             db "Reading... ", 0
3459 000145DC 2E2000              <1>
3460                               <1> msg_writing:
3461 000145DF 57726974696E672E2E- <1>             db "Writing... ", 0
3461 000145E8 2E2000              <1>
3462                               <1> percentagestr:
3463 000145EB 2020202500          <1>             db "   %", 0  ; "  0%" .. "100%"
3464                               <1> ; 11/04/2016
3465                               <1> Msg_No_Set_Space:
3466 000145F0 496E73756666696369- <1>                 db "Insufficient environment space!"
3466 000145F9 656E7420656E766972- <1>
3466 00014602 6F6E6D656E74207370- <1>
3466 0001460B 61636521            <1>
3467 0001460F 0D0A00              <1>                 db 0Dh, 0Ah, 0
3468                               <1> ; 18/04/2016
3469                               <1> isc_msg:
3470 00014612 0D0A                <1>             db 0Dh, 0Ah
3471 00014614 494E56414C49442053- <1>             db "INVALID SYSTEM CALL", 0
3471 0001461D 595354454D2043414C- <1>
3471 00014626 4C00                <1>
3472                               <1> usi_msg:
3473 00014628 0D0A                <1>             db 0Dh, 0Ah
3474 0001462A 554E444546494E4544- <1>             db "UNDEFINED SOFTWARE INTERRUPT", 0
3474 00014633 20534F465457415245- <1>
3474 0001463C 20494E544552525550- <1>
3474 00014645 5400                <1>
3475                               <1> ifc_msg:
3476 00014647 0D0A                <1>             db 0Dh, 0Ah
3477 00014649 494E56414C49442046- <1>             db "INVALID FUNCTION CALL"
3477 00014652 554E4354494F4E2043- <1>
3477 0001465B 414C4C              <1>
3478                               <1> inv_msg_for_trdos_v2:
3479 0001465E 20                  <1>             db 20h
3480 0001465F 666F72205452444F53- <1>             db "for TRDOS v2!"
3480 00014668 20763221            <1>
3481 0001466C 07                  <1>             db 07h
3482 0001466D 0D0A                <1>             db 0Dh, 0Ah
3483 0001466F 0D0A                <1>             db 0Dh, 0Ah
3484 00014671 494E5420            <1>             db "INT "
3485 00014675 303068              <1> int_num_str: db "00h"
3486 00014678 0D0A                <1>             db 0Dh, 0Ah
3487 0001467A 454158203A20        <1>             db "EAX : "
3488 00014680 303030303030303068- <1> eax_str:     db "00000000h", 0Dh, 0Ah
3488 00014689 0D0A                <1>
3489 0001468B 454950203A20        <1>             db "EIP : "
3490 00014691 303030303030303068- <1> eip_str:     db "00000000h", 0Dh, 0Ah, 0
3490 0001469A 0D0A00              <1>
3491                               <1>
3492                               <1> ; 07/10/2016
3493                               <1> ; Device names & parameters (for kernel devices)
3494                               <1>
3495 0001469D 90                  <1> align 2
3496                               <1> KDEV_NAME:
3497 0001469E 5454590000000000    <1>             db 'TTY',0,0,0,0,0 ; 1
3498 000146A6 4D454D0000000000    <1>             db 'MEM',0,0,0,0,0 ; 2
3499 000146AE 4644300000000000    <1>             db 'FD0',0,0,0,0,0 ; 3
3500 000146B6 4644310000000000    <1>             db 'FD1',0,0,0,0,0 ; 4
3501 000146BE 4844300000000000    <1>             db 'HD0',0,0,0,0,0 ; 5
3502 000146C6 4844310000000000    <1>             db 'HD1',0,0,0,0,0 ; 6
3503 000146CE 4844320000000000    <1>             db 'HD2',0,0,0,0,0 ; 7
3504 000146D6 4844330000000000    <1>             db 'HD3',0,0,0,0,0 ; 8
3505 000146DE 4C50540000000000    <1>             db 'LPT',0,0,0,0,0 ; 9
3506 000146E6 5454593000000000    <1>             db 'TTY0',0,0,0,0 ; 10
3507 000146EE 5454593100000000    <1>             db 'TTY1',0,0,0,0 ; 11
3508 000146F6 5454593200000000    <1>             db 'TTY2',0,0,0,0 ; 12
3509 000146FE 5454593300000000    <1>             db 'TTY3',0,0,0,0 ; 13
3510 00014706 5454593400000000    <1>             db 'TTY4',0,0,0,0 ; 14
3511 0001470E 5454593500000000    <1>             db 'TTY5',0,0,0,0 ; 15
3512 00014716 5454593600000000    <1>             db 'TTY6',0,0,0,0 ; 16
3513 0001471E 5454593700000000    <1>             db 'TTY7',0,0,0,0 ; 17
3514 00014726 5454593800000000    <1>             db 'TTY8',0,0,0,0 ; 18
3515 0001472E 5454593900000000    <1>             db 'TTY9',0,0,0,0 ; 19
3516 00014736 434F4D3100000000    <1>             db 'COM1',0,0,0,0 ; 18
3517 0001473E 434F4D3200000000    <1>             db 'COM2',0,0,0,0 ; 19
3518                               <1>             ;db 'CONSOLE',0      ; 1
3519                               <1>             ;db 'PRINTER',0   ; 9
3520                               <1>             ;db 'CDROM'    ; 20
3521                               <1>             ;db 'CDROM0'   ; 20
3522                               <1>             ;db 'CDROM1'   ; 21
3523                               <1>             ;db 'DVD'      ; 22
3524                               <1>             ;db 'DVD0'     ; 22
3525                               <1>             ;db 'DVD1'     ; 23
```

```
                                <1>             ;db 'USB'      ; 24
                                <1>             ;db 'USB0'     ; 24
                                <1>             ;db 'USB1'     ; 25
                                <1>             ;db 'USB2'     ; 26
                                <1>             ;db 'USB3'       ; 27
                                <1>             ;db 'KEYBOARD'       ; 1
                                <1>             ;db 'MOUSE'    ; 28
                                <1>             ;db 'SOUND'    ; 29
                                <1>             ;db 'VGA',0,0,0,0 ; 30
                                <1>             ;db 'CGA',0,0,0,0 ; 31
                                <1>             ;db 'AUDIO',0,0,0 ; 29
                                <1>             ;db 'VIDEO',0,0,0 ; 32
                                <1>             ;db 'MUSIC',0,0,0 ; 33
                                <1>             ;db 'ETHERNET'       ; 34
                                <1>             ;db 'SD0',0,0,0,0,0 ; 35
                                <1>             ;db 'SD1',0,0,0,0,0 ; 36
                                <1>             ;db 'SD2',0,0,0,0,0 ; 37
                                <1>             ;db 'SD3',0,0,0,0,0 ; 38
                                <1>             ;db 'SATA0'    ; 35
                                <1>             ;db 'SATA1'    ; 36
                                <1>             ;db 'SATA2'       ; 37
                                <1>             ;db 'SATA3'       ; 38
                                <1>             ;db 'PATA0',0,0,0  ; 5
                                <1>             ;db 'PATA1',0,0,0  ; 6
                                <1>             ;db 'PATA2',0,0,0  ; 7
                                <1>             ;db 'PATA3',0,0,0  ; 8
                                <1>             ;db 'WIRELESS'       ; 39
                                <1>             ;db 'HDMI',0,0,0,0 ; 40
00014746 4E554C4C00000000       <1>             db 'NULL',0,0,0,0 ; 0
                                <1>
                                <1> NumOfKernelDevNames equ ($-KDEV_NAME) / 8 ; 20 (07/10/2016)
                                <1>
                                <1> KDEV_NUMBER:
0001474E 010203040506070809     <1>             db 1,2,3,4,5,6,7,8,9
00014757 0A0B0C0D0E0F101112-    <1>             db 10,11,12,13,14,15,16,17,18,19
00014760 13                     <1>
00014761 121300                 <1>             db 18,19,0
                                <1>
                                <1> NumOfKernelDevices equ $ - KDEV_NUMBER
                                <1>
                                <1> KDEV_OADDR:
00014764 [813F0100]             <1>             dd otty ;tty  ; 1
00014768 [813F0100]             <1>             dd sret ;mem  ; 2
0001476C [813F0100]             <1>             dd sret ;fd0  ; 3
00014770 [813F0100]             <1>             dd sret ;fd1  ; 4
00014774 [813F0100]             <1>             dd sret ;hd0  ; 5
00014778 [813F0100]             <1>             dd sret ;hd1  ; 6
0001477C [813F0100]             <1>             dd sret ;hd2  ; 7
00014780 [813F0100]             <1>             dd sret ;hd3  ; 8
00014784 [813F0100]             <1>             dd sret ;lpt  ; 9
00014788 [813F0100]             <1>             dd ocvt ;tty0 ; 10
0001478C [813F0100]             <1>             dd ocvt ;tty1 ; 11
00014790 [813F0100]             <1>             dd ocvt ;tty2 ; 12
00014794 [813F0100]             <1>             dd ocvt ;tty3 ; 13
00014798 [813F0100]             <1>             dd ocvt ;tty4 ; 14
0001479C [813F0100]             <1>             dd ocvt ;tty5 ; 15
000147A0 [813F0100]             <1>             dd ocvt ;tty6 ; 16
000147A4 [813F0100]             <1>             dd ocvt ;tty7 ; 17
000147A8 [813F0100]             <1>             dd ocvt ;tty8 ; 18
000147AC [813F0100]             <1>             dd ocvt ;tty9 ; 19
                                <1>             ;dd ocvt ;com1 ; 18
                                <1>             ;dd ocvt ;com2 ; 19
000147B0 [813F0100]             <1>             dd sret ;null ; 20
                                <1> KDEV_CADDR:
000147B4 [813F0100]             <1>             dd ctty ;tty  ; 1
000147B8 [813F0100]             <1>             dd cret ;mem  ; 2
000147BC [813F0100]             <1>             dd cret ;fd0  ; 3
000147C0 [813F0100]             <1>             dd cret ;fd1  ; 4
000147C4 [813F0100]             <1>             dd cret ;hd0  ; 5
000147C8 [813F0100]             <1>             dd cret ;hd1  ; 6
000147CC [813F0100]             <1>             dd cret ;hd2  ; 7
000147D0 [813F0100]             <1>             dd cret ;hd3  ; 8
000147D4 [813F0100]             <1>             dd cret ;lpt  ; 9
000147D8 [813F0100]             <1>             dd ocvt ;tty0 ; 10
000147DC [813F0100]             <1>             dd ccvt ;tty1 ; 11
000147E0 [813F0100]             <1>             dd ccvt ;tty2 ; 12
000147E4 [813F0100]             <1>             dd ccvt ;tty3 ; 13
000147E8 [813F0100]             <1>             dd ccvt ;tty4 ; 14
000147EC [813F0100]             <1>             dd ccvt ;tty5 ; 15
000147F0 [813F0100]             <1>             dd ccvt ;tty6 ; 16
000147F4 [813F0100]             <1>             dd ccvt ;tty7 ; 17
000147F8 [813F0100]             <1>             dd ccvt ;tty8 ; 18
000147FC [813F0100]             <1>             dd ccvt ;tty9 ; 19
                                <1>             ;dd ccvt ;com1 ; 18
                                <1>             ;dd ccvt ;com2 ; 19
00014800 [813F0100]             <1>             dd cret ;null ; 20
                                <1>
                                <1> KDEV_RADDR:
00014804 [813F0100]             <1>             dd rtty ;tty  ; 1
00014808 [813F0100]             <1>             dd rmem ;mem  ; 2
0001480C [813F0100]             <1>             dd rfd  ;fd0  ; 3
00014810 [813F0100]             <1>             dd rfd  ;fd1  ; 4
00014814 [813F0100]             <1>             dd rhd  ;hd0  ; 5
00014818 [813F0100]             <1>             dd rhd  ;hd1  ; 6
0001481C [813F0100]             <1>             dd rhd  ;hd2  ; 7
00014820 [813F0100]             <1>             dd rhd  ;hd3  ; 8
00014824 [813F0100]             <1>             dd rlpt ;lpt  ; 9
00014828 [813F0100]             <1>             dd rcvt ;tty0 ; 10
0001482C [813F0100]             <1>             dd rcvt ;tty1 ; 11
00014830 [813F0100]             <1>             dd rcvt ;tty2 ; 12
00014834 [813F0100]             <1>             dd rcvt ;tty3 ; 13
00014838 [813F0100]             <1>             dd rcvt ;tty4 ; 14
0001483C [813F0100]             <1>             dd rcvt ;tty5 ; 15
00014840 [813F0100]             <1>             dd rcvt ;tty6 ; 16
00014844 [813F0100]             <1>             dd rcvt ;tty7 ; 17
```

```
3630 00014848 [813F0100]          <1>             dd rcvt ;tty8 ; 18
3631 0001484C [813F0100]          <1>             dd rcvt ;tty9 ; 19
3632                              <1>             ;dd rcvt ;com1 ; 18
3633                              <1>             ;dd rcvt ;com2 ; 19
3634 00014850 [65330100]          <1>             dd rnull ;null ; 20
3635                              <1> KDEV_WADDR:
3636 00014854 [813F0100]          <1>             dd wtty ;tty  ; 1
3637 00014858 [813F0100]          <1>             dd wmem ;mem  ; 2
3638 0001485C [813F0100]          <1>             dd wfd  ;fd0  ; 3
3639 00014860 [813F0100]          <1>             dd wfd  ;fd1  ; 4
3640 00014864 [813F0100]          <1>             dd whd  ;hd0  ; 5
3641 00014868 [813F0100]          <1>             dd whd  ;hd1  ; 6
3642 0001486C [813F0100]          <1>             dd whd  ;hd2  ; 7
3643 00014870 [813F0100]          <1>             dd whd  ;hd3  ; 8
3644 00014874 [813F0100]          <1>             dd wlpt ;lpt  ; 9
3645 00014878 [813F0100]          <1>             dd xmtt ;tty0 ; 10
3646 0001487C [813F0100]          <1>             dd xmtt ;tty1 ; 11
3647 00014880 [813F0100]          <1>             dd xmtt ;tty2 ; 12
3648 00014884 [813F0100]          <1>             dd xmtt ;tty3 ; 13
3649 00014888 [813F0100]          <1>             dd xmtt ;tty4 ; 14
3650 0001488C [813F0100]          <1>             dd xmtt ;tty5 ; 15
3651 00014890 [813F0100]          <1>             dd xmtt ;tty6 ; 16
3652 00014894 [813F0100]          <1>             dd xmtt ;tty7 ; 17
3653 00014898 [813F0100]          <1>             dd xmtt ;tty8 ; 18
3654 0001489C [813F0100]          <1>             dd xmtt ;tty9 ; 19
3655                              <1>             ;dd xmtt ;com1 ; 18
3656                              <1>             ;dd xmtt ;com2 ; 19
3657 000148A0 [66330100]          <1>             dd wnull ;null ; 20
3658                              <1>
3659                              <1> ; DEV_ACCESS bits:
3660                              <1>       ; bit 0 = accessable by normal users
3661                              <1>       ; bit 1 = read access permission
3662                              <1>       ; bit 2 = write access permission
3663                              <1>       ; bit 3 = IOCTL permission to users
3664                              <1>       ; bit 4 = block device if it is set
3665                              <1>       ; bit 5 = 16 bit or 1024 byte data
3666                              <1>       ; bit 6 = 32 bit or 2048 byte data
3667                              <1>       ; bit 7 = installable device driver
3668                              <1>
3669                              <1> KDEV_ACCESS: ; 08/10/2016
3670 000148A4 07                 <1>             db  00000111b; tty, 1
3671 000148A5 07                 <1>             db  00000111b; mem, 2
3672 000148A6 8F                 <1>             db  10001111b; fd0, 3
3673 000148A7 8F                 <1>             db  10001111b; fd1, 4
3674 000148A8 8F                 <1>             db  10001111b; hd0, 5
3675 000148A9 8F                 <1>             db  10001111b; hd1, 6
3676 000148AA 8F                 <1>             db  10001111b; hd2, 7
3677 000148AB 8F                 <1>             db  10001111b; hd3, 8
3678 000148AC 07                 <1>             db  00000111b   ; lpt, 9
3679 000148AD 07                 <1>             db  00000111b; tty0, 10
3680 000148AE 07                 <1>             db  00000111b; tty1, 11
3681 000148AF 07                 <1>             db  00000111b; tty2, 12
3682 000148B0 07                 <1>             db  00000111b; tty3, 13
3683 000148B1 07                 <1>             db  00000111b; tty4, 14
3684 000148B2 07                 <1>             db  00000111b; tty5, 15
3685 000148B3 07                 <1>             db  00000111b; tty6, 16
3686 000148B4 07                 <1>             db  00000111b; tty7, 17
3687 000148B5 07                 <1>             db  00000111b; tty8, 18
3688 000148B6 07                 <1>             db  00000111b; tty9, 19
3689                              <1>             ;db 00000111b; com1, 18
3690                              <1>             ;db 00000111b; com2, 19
3691 000148B7 00                 <1>             db  00000000b   ; null, 0
3692                              <1>
3693                              <1> ; 07/10/2016
3694                              <1> NumOfInstallableDevices equ 8
3695                              <1> NUMIDEV             equ NumOfInstallableDevices ; 8
3696                              <1> NUMOFDEVICES equ NumOfKernelDevices + NumOfInstallableDevices
3697                              <1>
3698                              <1> ; 26/02/2017
3699                              <1> ; IRQ Callback (& Signal Response Byte) service availibity
3700                              <1> ; 'syscalbac'
3701                              <1> ; ************************************************
3702                              <1> ; IRQ 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
3703                              <1> ; --- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
3704                              <1> ; --- 00 00 00 01 02 03 00 04 00 05 06 07 08 09 00 00
3705                              <1> ; ************************************************
3706                              <1> IRQenum:
3707 000148B8 00000010203000400- <1>       db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
3707 000148C1 05060708090000     <1>
3708                              <1>
3709                              <1> ; 28/08/2017
3710                              <1> ; 20/08/2017
3711                              <1> ; DMA Registers (for 'sysdma')
3712                              <1> ; 02/07/2017 (sb16mod.s)
3713 000148C8 00020406C0C4C8CC    <1> dma_adr:    db 0,2,4,6,0C0h,0C4h,0C8h,0CCh
3714 000148D0 01030507C2C6CACE    <1> dma_cnt:    db 1,3,5,7,0C2h,0C6h,0CAh,0CEh
3715 000148D8 878381828F8B898A    <1> dma_page:   db 87h,83h,81h,82h,8Fh,8Bh,89h,8Ah ; 03/08/2017
3716 000148E0 0A0A0A0AD4D4D4D4    <1> dma_mask:   db 0Ah,0Ah,0Ah,0Ah,0D4h,0D4h,0D4h,0D4h
3717 000148E8 0B0B0B0BD6D6D6D6    <1> dma_mod:    db 0Bh,0Bh,0Bh,0Bh,0D6h,0D6h,0D6h,0D6h
3718 000148F0 0C0C0C0CD8D8D8D8    <1> dma_flip:   db 0Ch,0Ch,0Ch,0Ch,0D8h,0D8h,0D8h,0D8h
3093
3094                                  ; 27/08/2014
3095                                  scr_row:
3096 000148F8 E0810B00                      dd 0B8000h + 0A0h + 0A0h + 0A0h ; Row 3
3097                                  scr_col:
3098 000148FC 00000000                      dd 0
3099
3100                                  Align 4
3101                                        ; 15/04/2016
3102                                        ; TRDOS 386 (TRDOS v2.0)
3103
3104                                        ; 21/08/2014
3105                                  ilist:
3106                                        ;times       32 dd cpu_except ; INT 0 to INT 1Fh
3107                                        ;
```

```
                                        ; Exception list
                                        ; 25/08/2014
00014900 [E00B0000]                     dd     exc0   ; 0h,  Divide-by-zero Error
00014904 [E70B0000]                     dd     exc1
00014908 [EE0B0000]                     dd     exc2
0001490C [F50B0000]                     dd     exc3
00014910 [F90B0000]                     dd     exc4
00014914 [FD0B0000]                     dd     exc5
00014918 [010C0000]                     dd     exc6   ; 06h,  Invalid Opcode
0001491C [050C0000]                     dd     exc7
00014920 [090C0000]                     dd     exc8
00014924 [0D0C0000]                     dd     exc9
00014928 [110C0000]                     dd     exc10
0001492C [150C0000]                     dd     exc11
00014930 [190C0000]                     dd     exc12
00014934 [1D0C0000]                     dd     exc13  ; 0Dh, General Protection Fault
00014938 [210C0000]                     dd     exc14  ; 0Eh, Page Fault
0001493C [250C0000]                     dd     exc15
00014940 [290C0000]                     dd     exc16
00014944 [2D0C0000]                     dd     exc17
00014948 [310C0000]                     dd     exc18
0001494C [350C0000]                     dd     exc19
00014950 [390C0000]                     dd     exc20
00014954 [3D0C0000]                     dd     exc21
00014958 [410C0000]                     dd     exc22
0001495C [450C0000]                     dd     exc23
00014960 [490C0000]                     dd     exc24
00014964 [4D0C0000]                     dd     exc25
00014968 [510C0000]                     dd     exc26
0001496C [550C0000]                     dd     exc27
00014970 [590C0000]                     dd     exc28
00014974 [5D0C0000]                     dd     exc29
00014978 [610C0000]                     dd     exc30
0001497C [650C0000]                     dd     exc31
                                      IRQ_list: ; 28/02/2017 ('syscalbac')
                                              ; Interrupt list
00014980 [54090000]                           dd     timer_int    ; INT 20h
                                              ;dd    irq0
00014984 [C6100000]                           dd     kb_int       ; 24/01/2016
                                              ;dd    irq1
00014988 [360B0000]                           dd     irq2
                                              ; COM2 int
0001498C [3A0B0000]                           dd     irq3
                                              ; COM1 int
00014990 [450B0000]                           dd     irq4
00014994 [500B0000]                           dd     irq5
                                      ;DISKETTE_INT: ;06/02/2015
00014998 [F3500000]                           dd     fdc_int             ; 16/02/2015, IRQ 6 handler
                                              ;dd    irq6
                                      ; Default IRQ 7 handler against spurious IRQs (from master PIC)
                                      ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
0001499C [BF0E0000]                           dd     default_irq7 ; 25/02/2015
                                              ;dd    irq7
                                      ; Real Time Clock Interrupt
000149A0 [BF0A0000]                           dd     rtc_int             ; 23/02/2015, IRQ 8 handler
                                              ;dd    irq8   ; INT 28h
000149A4 [600B0000]                           dd     irq9
000149A8 [640B0000]                           dd     irq10
000149AC [680B0000]                           dd     irq11
000149B0 [6C0B0000]                           dd     irq12
000149B4 [700B0000]                           dd     irq13
                                      ;HDISK_INT1:  ;06/02/2015
000149B8 [605A0000]                           dd     hdc1_int     ; 21/02/2015, IRQ 14 handler
                                              ;dd    irq14
                                      ;HDISK_INT2:  ;06/02/2015
000149BC [835A0000]                           dd     hdc2_int     ; 21/02/2015, IRQ 15 handler
                                              ;dd    irq15  ; INT 2Fh
                                              ; 14/08/2015
                                              ;dd    sysent       ; INT 30h (system calls)

                                              ; 15/04/2016
                                              ; TRDOS 386(TRDOS v2.0) Software Interrupts

000149C0 [1D4A0100]                           dd     int30h       ; Reserved for
                                                                    ; !!! Retro UNIX (RUNIX) !!!
                                                                    ; !!! SINGLIX !!! System Calls
000149C4 [5B170000]                           dd     int31h       ; Video BIOS (IBM PC/AT, Int 10h)
000149C8 [EB0E0000]                           dd     int32h       ; Keyboard Functions (IBM PC/AT, Int 16h)
000149CC [9F510000]                           dd     int33h       ; DISK I/O (IBM PC/AT, Int 13h)
000149D0 [182C0100]                           dd     int34h       ; #IOCTL# (I/O port access support for ring 3)
000149D4 [12690000]                           dd     int35h       ; Time/Date Functions (IBM PC/AT, Int 1Ah)
000149D8 [730D0000]                           dd     ignore_int   ; INT 36h : Timer Functions
000149DC [730D0000]                           dd     ignore_int   ; INT 37h
000149E0 [730D0000]                           dd     ignore_int   ; INT 38h
000149E4 [730D0000]                           dd     ignore_int   ; INT 39h
000149E8 [730D0000]                           dd     ignore_int   ; INT 3Ah
000149EC [730D0000]                           dd     ignore_int   ; INT 3Bh
000149F0 [730D0000]                           dd     ignore_int   ; INT 3Ch
000149F4 [730D0000]                           dd     ignore_int   ; INT 3Dh
000149F8 [730D0000]                           dd     ignore_int   ; INT 3Eh
000149FC [730D0000]                           dd     ignore_int   ; INT 3Fh
00014A00 [7AD60000]                           dd     sysent       ; INT 40h : !!! TRDOS 386 System Calls !!!
                                              ;dd    ignore_int
00014A04 00000000                             dd     0

                                      ; 20/08/2014
                                        ; /* This is the default interrupt "handler" :-) */
                                        ; Linux v0.12 (head.s)
                                      int_msg:
00014A08 556E6B6E6F776E2069-                  db "Unknown interrupt ! ", 0
00014A11 6E7465727275707420-
00014A1A 212000

                                      ; 15/04/2016
                                      ; TRDOS 386 (TRDOS v2.0)
```

```
3211
3212                                    ; 29/04/2016
3213                                    int30h:
3214                                    trdos_isc_routine:
3215                                          ; 02/05/2016
3216                                          ; 01/05/2016
3217                                          ; 29/04/2016
3218                                          ; 18/04/2016
3219                                          ; 15/04/2016 (TRDOS 386 = TRDOS v2.0)
3220                                          ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
3221                                          ; 03/02/2011 ('trdos_ifc_routine')
3222                                          ;
3223 00014A1D 8B1C24                        mov   ebx, [esp] ; EIP (next)
3224 00014A20 83EB02                        sub   ebx, 2 ; EIP (CD ##h)
3225
3226 00014A23 89C1                          mov   ecx, eax
3227 00014A25 8A4301                        mov   al, [ebx+1] ; CDh ##h
3228
3229 00014A28 66BA1000                      mov   dx, KDATA
3230 00014A2C 8EDA                          mov   ds, dx
3231 00014A2E 8EC2                          mov   es, dx
3232
3233 00014A30 FC                            cld
3234 00014A31 8B15[F0880100]                mov   edx, [k_page_dir]
3235 00014A37 0F22DA                        mov   cr3, edx
3236
3237 00014A3A E8F1F7FEFF                    call  bytetohex
3238 00014A3F 66A3[75460100]                mov   [int_num_str], ax
3239
3240 00014A45 89D8                          mov   eax, ebx ; EIP
3241 00014A47 E824F8FEFF                    call  dwordtohex
3242 00014A4C 8915[91460100]                mov   [eip_str], edx
3243 00014A52 A3[95460100]                  mov   [eip_str+4], eax
3244
3245 00014A57 89C8                          mov   eax, ecx
3246 00014A59 E812F8FEFF                    call  dwordtohex
3247 00014A5E 8915[80460100]                mov   [eax_str], edx
3248 00014A64 A3[84460100]                  mov   [eax_str+4], eax
3249
3250 00014A69 43                            inc   ebx
3251 00014A6A 8A03                          mov   al, [ebx] ; Interrupt number
3252
3253                                    trdos_isc_handler:
3254 00014A6C 80FE30                        cmp   dh, 30h ; Retro UNIX, SINGLIX System calls
3255 00014A6F 7507                          jne   short trdos_usi_handler
3256 00014A71 BE[12460100]                  mov   esi, isc_msg
3257 00014A76 EB05                          jmp   short loc_write_inv_system_call_msg
3258
3259                                    trdos_usi_handler:
3260 00014A78 BE[28460100]                  mov   esi, usi_msg
3261
3262                                    loc_write_inv_system_call_msg:
3263 00014A7D E8A329FFFF                    call  print_msg
3264                                        ; 29/04/2016
3265 00014A82 BE[5E460100]                  mov   esi, inv_msg_for_trdos_v2
3266 00014A87 E89929FFFF                    call  print_msg
3267
3268                                    loc_ifc_terminate_process:
3269                                        ; u.uno = process number
3270                                        ; 29/04/2016
3271
3272                                        ; 02/05/2016
3273 00014A8C FE05[5B030300]                inc   byte [sysflg] ; 0FFh -> 0
3274
3275 00014A92 B801000000                    mov   eax, 1
3276 00014A97 E9B78EFFFF                    jmp   sysexit
3277
3278                                    ; 07/03/2015
3279                                    ; Temporary Code
3280                                    display_disks:
3281 00014A9C 803D[CE6C0000]00              cmp   byte [fd0_type], 0
3282 00014AA3 7605                          jna   short ddsks1
3283 00014AA5 E87D000000                    call  pdskm
3284                                    ddsks1:
3285 00014AAA 803D[CF6C0000]00              cmp   byte [fd1_type], 0
3286 00014AB1 760C                          jna   short ddsks2
3287 00014AB3 C605[F74B0100]31              mov   byte [dskx], '1'
3288 00014ABA E868000000                    call  pdskm
3289                                    ddsks2:
3290 00014ABF 803D[D06C0000]00              cmp   byte [hd0_type], 0
3291 00014AC6 7654                          jna   short ddsk6
3292 00014AC8 66C705[F54B0100]68-           mov   word [dsktype], 'hd'
3292 00014AD0 64
3293 00014AD1 C605[F74B0100]30              mov   byte [dskx], '0'
3294 00014AD8 E84A000000                    call  pdskm
3295                                    ddsks3:
3296 00014ADD 803D[D16C0000]00              cmp   byte [hd1_type], 0
3297 00014AE4 7636                          jna   short ddsk6
3298 00014AE6 C605[F74B0100]31              mov   byte [dskx], '1'
3299 00014AED E835000000                    call  pdskm
3300                                    ddsks4:
3301 00014AF2 803D[D26C0000]00              cmp   byte [hd2_type], 0
3302 00014AF9 7621                          jna   short ddsk6
3303 00014AFB C605[F74B0100]32              mov   byte [dskx], '2'
3304 00014B02 E820000000                    call  pdskm
3305                                    ddsks5:
3306 00014B07 803D[D36C0000]00              cmp   byte [hd3_type], 0
3307 00014B0E 760C                          jna   short ddsk6
3308 00014B10 C605[F74B0100]33              mov   byte [dskx], '3'
3309 00014B17 E80B000000                    call  pdskm
3310                                    ddsk6:
3311 00014B1C BE[1F4C0100]                  mov   esi, nextline
3312 00014B21 E806000000                    call  pdskml
3313                                    pdskm_ok:
3314 00014B26 C3                            retn
```

```
                                    pdskm:
00014B27 BE[F34B0100]                       mov    esi, dsk_ready_msg
                                    pdskml:
00014B2C AC                                 lodsb
00014B2D 08C0                               or     al, al
00014B2F 74F5                               jz     short pdskm_ok
00014B31 56                                 push   esi
                                            ; 13/05/2016
00014B32 BB07000000                           mov     ebx, 7  ; Black background,
                                                      ; light gray forecolor
                                                      ; Video page 0 (bh=0)
00014B37 E846D7FEFF                         call   _write_tty
00014B3C 5E                                 pop    esi
00014B3D EBED                               jmp    short pdskml

00014B3F 90                         Align 2
                                            ; 21/08/2014
                                    exc_msg:
00014B40 435055206578636570-                db "CPU exception ! "
00014B49 74696F6E202120
                                    excnstr:           ; 25/08/2014
00014B50 3F3F68202045495020-                db "??h", "  EIP : "
00014B59 3A20
                                    EIPstr: ; 29/08/2014
00014B5B 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:
00014B67 07                                 db     07h
00014B68 0D0A                               db     0Dh, 0Ah
                                            ;db    "MEMORY ALLOCATION INFO", 0Dh, 0Ah, 0Dh, 0Ah
00014B6A 546F74616C206D656D-                db     "Total memory : "
00014B73 6F7279203A20
                                    mem_total_b_str: ; 10 digits
00014B79 303030303030303030-                db     "0000000000 bytes", 0Dh, 0Ah
00014B82 302062797465730D0A
00014B8B 202020202020202020-                db     "               ", 20h, 20h, 20h
00014B94 202020202020202020
                                    mem_total_p_str: ; 7 digits
00014B9D 303030303030302070-                db     "0000000 pages", 0Dh, 0Ah
00014BA6 616765730D0A
00014BAC 0D0A                               db     0Dh, 0Ah
00014BAE 46726565206D656D6F-                db     "Free memory  : "
00014BB7 727920203A20
                                    free_mem_b_str:  ; 10 digits
00014BBD 3F3F3F3F3F3F3F3F3F-                db     "?????????? bytes", 0Dh, 0Ah
00014BC6 3F2062797465730D0A
00014BCF 202020202020202020-                db     "               ", 20h, 20h, 20h
00014BD8 202020202020202020
                                    free_mem_p_str:  ; 7 digits
00014BE1 3F3F3F3F3F3F3F2070-                db     "??????? pages", 0Dh, 0Ah
00014BEA 616765730D0A
00014BF0 0D0A00                             db     0Dh, 0Ah, 0

                                    dsk_ready_msg:
00014BF3 0D0A                               db     0Dh, 0Ah
                                    dsktype:
00014BF5 6664                               db     'fd'
                                    dskx:
00014BF7 30                                 db     '0'
00014BF8 20                                 db     20h
00014BF9 697320524541445920-                db     'is READY ...'
00014C02 2E2E2E
00014C05 00                                 db     0

                                    setup_error_msg:
00014C06 0D0A                               db     0Dh, 0Ah
00014C08 4469736B2053657475-                db     'Disk Setup Error !'
00014C11 70204572726F722021
00014C1A 0D0A00                             db     0Dh, 0Ah,0

                                    next2line: ; 08/02/2016
00014C1D 0D0A                               db     0Dh, 0Ah
                                    nextline:
00014C1F 0D0A00                             db     0Dh, 0Ah, 0

                                    ; temporary
                                    ; 19/12/2020
                                    msg_lfb_addr:
                                            ;db    0Dh, 0Ah
00014C22 4C696E656172206672-                db     "Linear frame buffer at "
00014C2B 616D65206275666665-
00014C34 7220617420
                                    lfb_addr_str: ; 8 (hex) digits
00014C39 303030303030303068-                db     "00000000h", 0Dh, 0Ah
00014C42 0D0A
00014C44 0D0A00                             db     0Dh, 0Ah, 0

                                    ; KERNEL - SYSINIT Messages
                                    ; 24/08/2015
                                    ; 13/04/2015 - (Retro UNIX 386 v1 Beginning)
                                    ; 14/07/2013
                                    ;kernel_init_err_msg:
                                    ;       db 0Dh, 0Ah
                                    ;       db 07h
                                    ;       db 'Kernel initialization ERROR !'
                                    ;       db 0Dh, 0Ah, 0
```

```
                                ;welcome_msg:
                                ;       db 0Dh, 0Ah
                                ;       db 07h
                                ;       db 'Welcome to TRDOS 386 Operating System !'
                                ;       db 0Dh, 0Ah
                                ;       db 'by Erdogan Tan - 31/12/2017 (v2.0.0)'
                                ;       db 0Dh, 0Ah, 0
                                
                                panic_msg:
00014C47 0D0A07                         db 0Dh, 0Ah, 07h
00014C4A 4552524F523A204B65-            db 'ERROR: Kernel Panic !'
00014C53 726E656C2050616E69-
00014C5C 632021
00014C5F 0D0A00                         db 0Dh, 0Ah, 0
                                
                                ;msgl_drv_not_ready:
                                ;       db 07h, 0Dh, 0Ah
                                ;         db 'Drive not ready or read error !'
                                ;         db 0Dh, 0Ah, 0
                                
                                starting_msg:
00014C62 5475726B6973682052-            db "Turkish Rational DOS v2.0 [12/04/2021] ...", 0
00014C6B 6174696F6E616C2044-
00014C74 4F532076322E30205B-
00014C7D 31322F30342F323032-
00014C86 315D202E2E2E00
                                NextLine:
00014C8D 0D0A00                         db 0Dh, 0Ah, 0
                                
                                %include 'audio.s' ; 03/04/2017
                              <1> ; ****************************************************************************
                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.2 - audio.s
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Last Update: 01/09/2020
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Beginning: 03/04/2017
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
                              <1> ; ****************************************************************************
                              <1>
                              <1> ; AUDIO CONTROLLER & CODEC DEFINITIONS & CODE FOR TRDOS 386
                              <1>
                              <1> ;============================================================================
                              <1> ;               EQUATES
                              <1> ;============================================================================
                              <1>
                              <1> ; PCI EQUATES
                              <1>
                              <1> BIT0  EQU 1
                              <1> BIT1  EQU 2
                              <1> BIT2  EQU 4
                              <1> BIT3  EQU 8
                              <1> BIT4  EQU 10h
                              <1> BIT5  EQU 20h
                              <1> BIT6  EQU 40h
                              <1> BIT7  EQU 80h
                              <1> BIT8  EQU 100h
                              <1> BIT9  EQU 200h
                              <1> BIT10 EQU 400h
                              <1> BIT11 EQU 800h
                              <1> BIT12 EQU 1000h
                              <1> BIT13 EQU 2000h
                              <1> BIT14 EQU 4000h
                              <1> BIT15 EQU 8000h
                              <1> BIT16 EQU 10000h
                              <1> BIT17 EQU 20000h
                              <1> BIT18 EQU 40000h
                              <1> BIT19 EQU 80000h
                              <1> BIT20 EQU 100000h
                              <1> BIT21 EQU 200000h
                              <1> BIT22 EQU 400000h
                              <1> BIT23 EQU 800000h
                              <1> BIT24 EQU 1000000h
                              <1> BIT25 EQU 2000000h
                              <1> BIT26 EQU 4000000h
                              <1> BIT27 EQU 8000000h
                              <1> BIT28 EQU 10000000h
                              <1> BIT29 EQU 20000000h
                              <1> BIT30 EQU 40000000h
                              <1> BIT31 EQU 80000000h
                              <1> NOT_BIT31 EQU 7FFFFFFFh
                              <1>
                              <1> ; PCI equates
                              <1> ; PCI function address (PFA)
                              <1> ; bit 31 = 1
                              <1> ; bit 23:16 = bus number     (0-255)
                              <1> ; bit 15:11 = device number  (0-31)
                              <1> ; bit 10:8 = function number (0-7)
                              <1> ; bit 7:0 = register number  (0-255)
                              <1>
                              <1> IO_ADDR_MASK    EQU     0FFFEh  ; mask off bit 0 for reading BARs
                              <1> PCI_INDEX_PORT  EQU     0CF8h
                              <1> PCI_DATA_PORT   EQU     0CFCh
                              <1> PCI32           EQU     BIT31   ; bitflag to signal 32bit access
                              <1> PCI16           EQU     BIT30   ; bitflag for 16bit access
                              <1> NOT_PCI32_PCI16    EQU    03FFFFFFFh ; NOT BIT31+BIT30 ; 19/03/2017
                              <1>
                              <1> PCI_FN0         EQU     0 << 8
                              <1> PCI_FN1         EQU     1 << 8
                              <1> PCI_FN2         EQU     2 << 8
                              <1> PCI_FN3         EQU     3 << 8
                              <1> PCI_FN4         EQU     4 << 8
                              <1> PCI_FN5         EQU     5 << 8
                              <1> PCI_FN6         EQU     6 << 8
```

```
                        <1> PCI_FN7         EQU     7 << 8
                        <1>
                        <1> PCI_CMD_REG  EQU    04h    ; reg 04, command reg
                        <1> IO_ENA              EQU    BIT0   ; i/o decode enable
                        <1> MEM_ENA             EQU    BIT1   ; memory decode enable
                        <1> BM_ENA              EQU     BIT2 ; bus master enable
                        <1>
                        <1> ; VIA VT8233 EQUATES
                        <1>
                        <1> VIA_VID             equ 1106h    ; VIA's PCI vendor ID
                        <1> VT8233_DID      equ 3059h ; VT8233 (VT8235) device ID
                        <1>
                        <1> PCI_IO_BASE          equ 10h
                        <1> AC97_INT_LINE        equ 3Ch
                        <1> VIA_ACLINK_CTRL      equ 41h
                        <1> VIA_ACLINK_STAT      equ 40h
                        <1> VIA_ACLINK_C00_READY equ 01h ; primary codec ready
                        <1>
                        <1> VIA_REG_AC97      equ 80h ; dword
                        <1>
                        <1> VIA_ACLINK_CTRL_ENABLE    equ   80h ; 0: disable, 1: enable
                        <1> VIA_ACLINK_CTRL_RESET     equ   40h ; 0: assert, 1: de-assert
                        <1> VIA_ACLINK_CTRL_SYNC      equ   20h ; 0: release SYNC, 1: force SYNC hi
                        <1> VIA_ACLINK_CTRL_VRA       equ   08h ; 0: disable VRA, 1: enable VRA
                        <1> VIA_ACLINK_CTRL_PCM       equ   04h ; 0: disable PCM, 1: enable PCM
                        <1>                                     ; 3D Audio Channel slots 3/4
                        <1> VIA_ACLINK_CTRL_INIT      equ  (VIA_ACLINK_CTRL_ENABLE +
VIA_ACLINK_CTRL_RESET +                         VIA_ACLINK_CTRL_PCM +                                   VIA_ACLINK_CTRL_VRA)
                        <1>
                        <1> CODEC_AUX_VOL             equ   04h
                        <1> VIA_REG_AC97_BUSY  equ   01000000h ;(1<<24)
                        <1> VIA_REG_AC97_CMD_SHIFT    equ   10h ; 16
                        <1> VIA_REG_AC97_PRIMARY_VALID equ 02000000h ;(1<<25)
                        <1> VIA_REG_AC97_READ  equ   00800000h ;(1<<23)
                        <1> VIA_REG_AC97_CODEC_ID_SHIFT   equ  1Eh ; 30
                        <1> VIA_REG_AC97_CODEC_ID_PRIMARY equ  0
                        <1> VIA_REG_AC97_DATA_SHIFT equ   0
                        <1> VIADEV_PLAYBACK         equ   0
                        <1> VIA_REG_OFFSET_STATUS   equ   0    ;; byte - channel status
                        <1> VIA_REG_OFFSET_CONTROL  equ   01h  ;; byte - channel control
                        <1> VIA_REG_CTRL_START equ   80h  ;; WO
                        <1> VIA_REG_CTRL_TERMINATE  equ   40h  ;; WO
                        <1> VIA_REG_CTRL_PAUSE      equ   08h  ;; RW
                        <1> VIA_REG_CTRL_RESET      equ   01h  ;; RW - probably reset? undocumented
                        <1> VIA_REG_OFFSET_STOP_IDX equ   08h  ;; dword - stop index, channel type, sample rate
                        <1> VIA8233_REG_TYPE_16BIT  equ   200000h ;; RW
                        <1> VIA8233_REG_TYPE_STEREO equ   100000h ;; RW
                        <1> VIA_REG_OFFSET_CURR_INDEX equ 0Fh ;; byte - channel current index (for via8233 only)
                        <1> VIA_REG_OFFSET_TABLE_PTR equ  04h  ;; dword - channel table pointer
                        <1> VIA_REG_OFFSET_CURR_PTR equ   04h  ;; dword - channel current pointer
                        <1> VIA_REG_OFS_PLAYBACK_VOLUME_L equ  02h ;; byte
                        <1> VIA_REG_OFS_PLAYBACK_VOLUME_R equ  03h ;; byte
                        <1> VIA_REG_CTRL_AUTOSTART    equ   20h
                        <1> VIA_REG_CTRL_INT_EOL      equ   02h
                        <1> VIA_REG_CTRL_INT_FLAG     equ   01h
                        <1> VIA_REG_CTRL_INT   equ  (VIA_REG_CTRL_INT_FLAG +                                  VIA_REG_CTRL_INT_EOL
+                   VIA_REG_CTRL_AUTOSTART)
                        <1>
                        <1> VIA_REG_STAT_STOP_IDX     equ   10h    ;; RO ; 27/07/2020
                        <1>                                  ; current index = stop index
                        <1> VIA_REG_STAT_STOPPED      equ   04h    ;; RWC
                        <1> VIA_REG_STAT_EOL   equ   02h    ;; RWC
                        <1> VIA_REG_STAT_FLAG  equ   01h    ;; RWC
                        <1> VIA_REG_STAT_ACTIVE       equ   80h    ;; RO
                        <1> ; 28/11/2016
                        <1> VIA_REG_STAT_LAST  equ   40h    ;; RO
                        <1> VIA_REG_STAT_TRIGGER_QUEUED equ 08h  ;; RO
                        <1> VIA_REG_CTRL_INT_STOP     equ   04h  ; Interrupt on Current Index = Stop Index
                        <1>                                ; and End of Block
                        <1>
                        <1> VIA_REG_OFFSET_CURR_COUNT equ 0Ch ;; dword - channel current count, index
                        <1>
                        <1> PORTB       EQU    061h
                        <1> REFRESH_STATUS     EQU    010h   ; Refresh signal status
                        <1>
                        <1> ; AC97 Codec registers.
                        <1>
                        <1> ; 22/07/2020
                        <1> ; REALTEK ALC655 and ADI SOUNDMAX AD1980 CODEC MIXER REGISTERS
                        <1>
                        <1> ; each codec/mixer register is 16bits
                        <1>
                        <1> CODEC_RESET_REG                 equ     00h  ; reset codec
                        <1> CODEC_MASTER_VOL_REG            equ     02h  ; master volume
                        <1> CODEC_HP_VOL_REG                equ     04h  ; headphone volume ; AD1980
                        <1> CODEC_MASTER_MONO_VOL_REG       equ     06h  ; master mono volume (mono-out)
                        <1> ;CODEC_MASTER_TONE_REG          equ     08h  ; master tone (R+L) ; (not used)
                        <1> CODEC_PCBEEP_VOL_REG            equ     0Ah  ; PC beep volume ; ALC655
                        <1> CODEC_PHONE_VOL_REG             equ     0Ch  ; phone volume
                        <1> CODEC_MIC_VOL_REG               equ     0Eh  ; mic volume
                        <1> CODEC_LINE_IN_VOL_REG           equ     10h  ; line in volume
                        <1> CODEC_CD_VOL_REG                equ     12h  ; CD volume
                        <1> ;CODEC_VID_VOL_REG              equ     14h  ; video volume ; (not used)
                        <1> CODEC_AUX_VOL_REG               equ     16h  ; aux volume
                        <1> CODEC_PCM_OUT_REG               equ     18h  ; PCM out volume
                        <1> CODEC_RECORD_SELECT_REG         equ     1Ah  ; record select
                        <1> CODEC_RECORD_VOL_REG            equ     1Ch  ; record volume (record gain)
                        <1> ;CODEC_RECORD_MIC_VOL_REG       equ     1Eh  ; record mic volume ; (not used)
                        <1> CODEC_GP_REG                    equ     20h  ; general purpose
                        <1> ;CODEC_3D_CONTROL_REG           equ     22h  ; 3D control
                        <1> ;;CODEC_AUDIO_INT_PAGING_REG    equ    24h   ; audio int & paging ; (not used)
                        <1> CODEC_POWER_CTRL_REG            equ     26h  ; power down control
                        <1> CODEC_EXT_AUDIO_REG             equ     28h  ; extended audio ID
                        <1> CODEC_EXT_AUDIO_CTRL_REG        equ     2Ah  ; extended audio status/control
                        <1> CODEC_PCM_FRONT_DACRATE_REG     equ     2Ch  ; PCM front sample rate
```

```
                                        <1> CODEC_PCM_SURND_DACRATE_REG     equ     2Eh  ; PCM surround sample rate
                                        <1> CODEC_PCM_LFE_DACRATE_REG       equ     30h  ; PCM Center/LFE sample rate
                                        <1> CODEC_LR_ADCRATE_REG            equ     32h  ; PCM input sample rate
                                        <1> CODEC_MIC_ADCRATE_REG           equ     34h  ; mic in sample rate  ; AD1980
                                        <1> CODEC_PCM_LFE_VOL_REG           equ     36h  ; PCM Center/LFE volume
                                        <1> CODEC_PCM_SURND_VOL_REG         equ     38h  ; PCM surround volume
                                        <1> ;CODEC_SPDIF_CTRL_REG           equ     3Ah  ; S/PDIF control
                                        <1> ; 22/07/2020
                                        <1> CODEC_MISC_CRTL_BITS_REG  equ    76h    ; misc control bits ; AD1980
                                        <1> ;
                                        <1> CODEC_VENDOR_ID1          equ    7Ch    ; REALTEK: 414Ch, ADI: 4144h
                                        <1> CODEC_VENDOR_ID2          equ    7Eh    ; REALTEK: 4760h, ADI: 5370h
                                        <1>
                                        <1> ; VT8233 SGD bits (21/04/2017)
                                        <1> FLAG  EQU BIT30
                                        <1> EOL   EQU BIT31
                                        <1>
                                        <1> ; INTEL ICH EQUATES
                                        <1> ; 28/05/2017
                                        <1> INTEL_VID    equ    8086h  ; Intel's PCI vendor ID
                                        <1> ICH_DID            equ    2415h  ; ICH (82801AA) device ID
                                        <1> NAMBAR_REG      equ       10h    ; native audio mixer Base Address Register
                                        <1> NABMBAR_REG     equ       14h    ; native audio bus mastering Base Addr Reg
                                        <1>
                                        <1> PI_CR_REG       equ     0Bh     ; PCM in Control Register
                                        <1> PO_CR_REG    equ     1Bh     ; PCM out Control Register
                                        <1> MC_CR_REG    equ     2Bh     ; MIC in Control Register
                                        <1>
                                        <1> PI_SR_REG    equ     6       ; PCM in Status register
                                        <1> PO_SR_REG    equ     16h     ; PCM out Status register
                                        <1> MC_SR_REG    equ     26h     ; MIC in Status register
                                        <1>
                                        <1> IOCE         equ     BIT4    ; interrupt on complete enable.
                                        <1> FEIFE        equ     BIT3    ; set if you want an interrupt to fire
                                        <1> LVBIE        equ     BIT2    ; last valid buffer interrupt enable.
                                        <1> RR           equ     BIT1    ; reset registers. Nukes all regs
                                        <1>                                  ; except bits 4:2 of this register.
                                        <1>                                  ; Only set this bit if BIT 0 is 0
                                        <1> RPBM         equ     BIT0    ; Run/Pause
                                        <1>                             ; set this bit to start the codec!
                                        <1>
                                        <1> PI_BDBAR_REG equ     0       ; PCM in buffer descriptor BAR
                                        <1> PO_BDBAR_REG equ     10h     ; PCM out buffer descriptor BAR
                                        <1> MC_BDBAR_REG equ     20h     ; MIC in buffer descriptor BAR
                                        <1>
                                        <1> PI_CIV_REG   equ     4       ; PCM in current Index value (RO)
                                        <1> PO_CIV_REG   equ     14h     ; PCM out current Index value (RO)
                                        <1> MC_CIV_REG   equ     24h     ; MIC in current Index value (RO)
                                        <1>
                                        <1> PI_LVI_REG   equ     5       ; PCM in Last Valid Index
                                        <1> PO_LVI_REG   equ     15h     ; PCM out Last Valid Index
                                        <1> MC_LVI_REG   equ     25h     ; MIC in Last Valid Index
                                        <1>
                                        <1> IOC          equ     BIT31; Fire an interrupt whenever this
                                        <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
00014C90 B886801524                     <1>       mov     eax, (ICH_DID << 16) + INTEL_VID
00014C95 E876000000                     <1>         call    pciFindDevice
00014C9A 730D                           <1>         jnc     short d_ac97_1
                                        <1> d_ac97_0:
                                        <1> ; couldn't find the audio device!
00014C9C 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
00014C9D B806115930                     <1>       mov     eax, (VT8233_DID << 16) + VIA_VID
00014CA2 E869000000                     <1>         call    pciFindDevice
                                        <1> ;       jnc     short d_vt8233_0
                                        <1> ; couldn't find the audio device!
                                        <1> ;     retn
00014CA7 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>
00014CA9 A3[789B0100]           <1>       mov   [audio_dev_id], eax
00014CAE 8915[7C9B0100]         <1>       mov   [audio_vendor], edx
                                <1>
                                <1>       ; init controller
00014CB4 B004                   <1>       mov   al, PCI_CMD_REG ; command register (04h)
00014CB6 E8E2000000             <1>       call  pciRegRead32
                                <1>
                                <1>       ; eax = BUS/DEV/FN/REG
                                <1>       ; edx = STATUS/COMMAND
                                <1>       ;     SSSSSSSSSSSSSSSSCCCCCCCCCCCCCCCC
00014CBB 8915[809B0100]         <1>       mov   [audio_stats_cmd], edx
                                <1>
00014CC1 B010                   <1>       mov   al, PCI_IO_BASE ; IO base address register (10h)
                                <1>       ;mov  al, NAMBAR_REG      ; Native Audio Mixer BAR (10h)
00014CC3 E8D5000000             <1>       call  pciRegRead32
                                <1>
00014CC8 66813D[7C9B0100]86-    <1>       cmp   word [audio_vendor], 8086h ; AC'97 ?
00014CD0 80                     <1>
00014CD1 751F                   <1>       jne   short d_vt8233_1
                                <1>
00014CD3 6683E2FE               <1>       and   dx, 0FFFEh ; Audio Codec IO_ADDR_MASK
00014CD7 668915[A89B0100]       <1>       mov   [NAMBAR], dx
                                <1>
00014CDE B014                   <1>       mov   al, NABMBAR_REG ; Native Audio Bus Mastering BAR (14h)
00014CE0 E8B8000000             <1>       call  pciRegRead32
                                <1>
00014CE5 6683E2C0               <1>       and   dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00014CE9 668915[AA9B0100]       <1>       mov   [NABMBAR], dx
                                <1>         ;mov [audio_io_base], dx
                                <1>
00014CF0 EB0B                   <1>       jmp   short d_ac97_2
                                <1>
                                <1> d_vt8233_1:
00014CF2 6683E2C0               <1>       and     dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00014CF6 668915[769B0100]       <1>         mov     [audio_io_base], dx
                                <1>
                                <1> d_ac97_2:
                                <1>       ; 10/06/2017
00014CFD B03C                   <1>       mov   al, AC97_INT_LINE ; Interrupt Line Register (3Ch)
                                <1>       ;call pciRegRead32
00014CFF E886000000             <1>       call  pciRegRead8
                                <1>
                                <1>       ;and  edx, 0FFh
00014D04 6681E2FF00             <1>       and   dx, 0FFh
                                <1>
00014D09 8815[739B0100]         <1>       mov     [audio_intr], dl
                                <1>
00014D0F C3                     <1>       retn
                                <1>
                                <1>       ;; (Note: Interrupts are already enabled by TRDOS 386 kernel!)
                                <1>       ;mov  cx, dx
                                <1>
                                <1>       ;in   al, 0A1h ; irq 8-15
                                <1>       ;mov  ah, al
                                <1>       ;in   al, 21h  ; irq 0-7
                                <1>       ;btr  ax, dx  ; unmask ; 17/03/2017
                                <1>       ;;bts ax, dx   ; MASK interrupt ; 10/06/2017
                                <1>       ;out  21h, al  ; irq <= 7
                                <1>       ;mov  al, ah
                                <1>       ;out  0A1h, al ; irq > 7
                                <1>       ;
                                <1>
                                <1>       ; 10/06/2017
                                <1>       ; === Intel ICH I/O Controller Hub Datasheet, Section 8.1.16 ===
                                <1>       ; PRQ[n]_ROUT Register (61h, PRQB) Bit 7:
                                <1>       ; Interrupt Routing Enable (IRQEN).
                                <1>       ; 0 = The corresponding PIRQ is routed to one of the ISA-compatible
                                <1>       ;     interrupts specified in bits[3:0].
                                <1>       ; 1 = The PIRQ is not routed to the 8259.
                                <1>       ; Note: If the PIRQ is intended to cause an interrupt to the ICH's
                                <1>       ;     integrated I/O APIC, then this bit should be set to 0 and
                                <1>       ;     the APIC_EN bit should be set to 1.
                                <1>       ;     The IRQEN must be set to 0 and the PIRQ routed to
                                <1>       ;     an 8259 interrupt via the IRQ Routing filed (bits[3:0).
                                <1>       ;     The corresponding 8259 interrupt must be masked via the
                                <1>       ;     appropriated bit in the 8259's OCW1 (Interrupt Mask)
                                <1>       ;     register. The IOAPIC must then be enabled by setting
                                <1>       ;     the APIC_EN bit in the GEN_CNTL register.
                                <1>
                                <1>       ;mov  eax, 0F861h  ; D31:F0
                                <1>             ;AL=61h : PIRQ[B] Routing Control Reg, LPC interface
                                <1>       ;;mov  dl, [audio_intr]
                                <1>       ;call  pciRegWrite8
                                <1>       ;;mov  al, 0D0h ; General Control Register (GEN_CTL)
                                <1>       ;;call pciRegRead32
                                <1>       ;;or   edx, 100h ; Bit 8, APIC_EN (Enable I/O APIC)
                                <1>       ;;;call      pciRegWrite32
                                <1>       ;;and  edx, ~100h
                                <1>       ;;call pciRegWrite32 ; ; Bit 8, APIC_EN (Disable I/O APIC)
                                <1>       ;
                                <1>
                                <1>       ;mov  dx, 4D1h     ; 8259 ELCR2
                                <1>       ;in   al, dx
                                <1>       ;mov  ah, al
                                <1>       ;;mov  dx, 4D0h     ; 8259 ELCR1
                                <1>       ;dec  dl
```

```
                        <1>       ;in    al, dx
                        <1>       ;bts   ax, cx
                        <1>       ;;mov  dx, 4D0h
                        <1>       ;out   dx, al       ; set level-triggered mode
                        <1>       ;mov   al, ah ; 29/05/2017
                        <1>       ;;mov  dx, 4D1h
                        <1>       ;inc   dl
                        <1>       ;out   dx, al       ; set level-triggered mode
                        <1>
                        <1>       ;xor   eax, eax ; 0
                        <1>
                        <1>       ;retn
                        <1>
                        <1> ; CODE for PCI
                        <1>
                        <1> pciFindDevice:
                        <1>       ; 03/04/2017 ('pci.asm', 20/03/2017)
                        <1>       ;
                        <1>       ; scan through PCI space looking for a device+vendor ID
                        <1>       ;
                        <1>       ; Entry: EAX=Device+Vendor ID
                        <1>       ;
                        <1>       ; Exit: EAX=PCI address if device found
                        <1>       ;       EDX=Device+Vendor ID
                        <1>       ;         CY clear if found, set if not found. EAX invalid if CY set.
                        <1>       ;
                        <1>       ; Destroys: ebx, esi, edi, cl
                        <1>       ;
                        <1>
                        <1>       ;push  ecx
00014D10 50             <1>       push   eax
                        <1>       ;push  esi
                        <1>       ;push  edi
                        <1>
00014D11 89C6           <1>         mov     esi, eax                ; save off vend+device ID
00014D13 BF00FFFF7F     <1>         mov     edi, (80000000h - 100h) ; start with bus 0, dev 0 func 0
                        <1>
                        <1> nextPCIdevice:
00014D18 81C700010000   <1>         add     edi, 100h
00014D1E 81FF00F8FF80   <1>         cmp     edi, 80FFF800h         ; scanned all devices?
00014D24 F9             <1>         stc
00014D25 740C           <1>         je      short PCIScanExit       ; not found
                        <1>
00014D27 89F8           <1>         mov     eax, edi                ; read PCI registers
00014D29 E86F000000     <1>         call    pciRegRead32
00014D2E 39F2           <1>         cmp     edx, esi                ; found device?
00014D30 75E6           <1>         jne     short nextPCIdevice
00014D32 F8             <1>         clc
                        <1>
                        <1> PCIScanExit:
00014D33 9C             <1>       pushf
00014D34 B8FFFFFF7F     <1>       mov    eax, NOT_BIT31      ; 19/03/2017
00014D39 21F8           <1>       and    eax, edi     ; return only bus/dev/fn #
00014D3B 9D             <1>       popf
                        <1>
                        <1>       ;pop   edi
                        <1>       ;pop   esi
00014D3C 5A             <1>       pop    edx
                        <1>       ;pop   ecx
00014D3D 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>
00014D3E 53             <1>       push   ebx
00014D3F 51             <1>       push   ecx
00014D40 89C3           <1>         mov     ebx, eax        ; save eax, dh
00014D42 88F1           <1>         mov     cl, dh
                        <1>
00014D44 25FFFFFF3F     <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
00014D49 0D00000080     <1>         or      eax, BIT31             ; make a PCI access request
00014D4E 24FC           <1>         and     al, ~3 ; NOT 3         ; force index to be dword
                        <1>
00014D50 66BAF80C       <1>         mov     dx, PCI_INDEX_PORT
00014D54 EF             <1>         out  dx, eax                   ; write PCI selector
                        <1>
00014D55 66BAFC0C       <1>         mov     dx, PCI_DATA_PORT
00014D59 88D8           <1>         mov     al, bl
00014D5B 2403           <1>         and     al, 3                  ; figure out which port to
00014D5D 00C2           <1>         add     dl, al                 ; read to
                        <1>
00014D5F F7C3000000C0   <1>       test    ebx, PCI32+PCI16
00014D65 7507           <1>         jnz     short _pregr0
00014D67 EC             <1>       in     al, dx              ; return 8 bits of data
00014D68 88C2           <1>         mov  dl, al
00014D6A 88CE           <1>       mov     dh, cl                   ; restore dh for 8 bit read
00014D6C EB12           <1>       jmp    short _pregr2
                        <1> _pregr0:
00014D6E F7C300000080   <1>       test    ebx, PCI32
```

```
3926 00014D74 7507              <1>         jnz short _pregr1
3927 00014D76 66ED              <1>       in    ax, dx
3928 00014D78 6689C2            <1>         mov     dx, ax                ; return 16 bits of data
3929 00014D7B EB03              <1>       jmp   short _pregr2
3930                            <1> _pregr1:
3931 00014D7D ED                <1>       in    eax, dx                   ; return 32 bits of data
3932 00014D7E 89C2              <1>       mov   edx, eax
3933                            <1> _pregr2:
3934 00014D80 89D8              <1>       mov     eax, ebx          ; restore eax
3935 00014D82 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
3936 00014D87 59                <1>       pop   ecx
3937 00014D88 5B                <1>       pop   ebx
3938 00014D89 C3                <1>       retn
3939                            <1>
3940                            <1> pciRegRead8:
3941 00014D8A 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit read size
3942 00014D8F EBAD              <1>         jmp     short pciRegRead; call generic PCI access
3943                            <1>
3944                            <1> pciRegRead16:
3945 00014D91 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit read size
3946 00014D96 0D00000040        <1>         or      eax, PCI16             ; call generic PCI access
3947 00014D9B EBA1              <1>         jmp     short pciRegRead
3948                            <1>
3949                            <1> pciRegRead32:
3950 00014D9D 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit read size
3951 00014DA2 0D00000080        <1>         or      eax, PCI32             ; call generic PCI access
3952 00014DA7 EB95              <1>         jmp     pciRegRead
3953                            <1>
3954                            <1> pciRegWrite:
3955                            <1>       ; 03/04/2017 ('pci.asm', 29/11/2016)
3956                            <1>       ;
3957                            <1>       ; 8/16/32bit PCI writer
3958                            <1>       ;
3959                            <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
3960                            <1>       ;           BIT31 set if 32 bit access requested
3961                            <1>       ;           BIT30 set if 16 bit access requested
3962                            <1>       ;           otherwise defaults to 8bit read
3963                            <1>       ;         DL/DX/EDX data to write depending on size
3964                            <1>       ;
3965                            <1>       ; Note1: this routine is meant to be called via pciRegWrite8,
3966                            <1>       ;       pciRegWrite16 or pciRegWrite32 as detailed below.
3967                            <1>       ;
3968                            <1>       ; Note2: don't attempt to write 32bits of data from a non dword
3969                            <1>       ;       aligned reg number. Likewise, don't do 16 bit writes from
3970                            <1>       ;       non word aligned reg #
3971                            <1>
3972 00014DA9 53                <1>       push  ebx
3973 00014DAA 51                <1>       push  ecx
3974 00014DAB 89C3              <1>         mov     ebx, eax        ; save eax, edx
3975 00014DAD 89D1              <1>         mov     ecx, edx
3976 00014DAF 25FFFFFF3F        <1>       and     eax, NOT_PCI32_PCI16     ; clear out data size request
3977 00014DB4 0D00000080        <1>         or      eax, BIT31             ; make a PCI access request
3978 00014DB9 24FC              <1>         and     al, ~3 ; NOT 3         ; force index to be dword
3979                            <1>
3980 00014DBB 66BAF80C          <1>         mov     dx, PCI_INDEX_PORT
3981 00014DBF EF                <1>         out  dx, eax                   ; write PCI selector
3982                            <1>
3983 00014DC0 66BAFC0C          <1>         mov     dx, PCI_DATA_PORT
3984 00014DC4 88D8              <1>         mov     al, bl
3985 00014DC6 2403              <1>         and     al, 3                  ; figure out which port to
3986 00014DC8 00C2              <1>         add     dl, al                 ; write to
3987                            <1>
3988 00014DCA F7C3000000C0      <1>       test    ebx, PCI32+PCI16
3989 00014DD0 7505              <1>         jnz     short _pregw0
3990 00014DD2 88C8              <1>       mov   al, cl                   ; put data into al
3991 00014DD4 EE                <1>       out   dx, al
3992 00014DD5 EB12              <1>       jmp   short _pregw2
3993                            <1> _pregw0:
3994 00014DD7 F7C300000080      <1>       test    ebx, PCI32
3995 00014DDD 7507              <1>         jnz     short _pregw1
3996 00014DDF 6689C8            <1>       mov   ax, cx              ; put data into ax
3997 00014DE2 66EF              <1>       out   dx, ax
3998 00014DE4 EB03              <1>       jmp   short _pregw2
3999                            <1> _pregw1:
4000 00014DE6 89C8              <1>       mov   eax, ecx            ; put data into eax
4001 00014DE8 EF                <1>       out   dx, eax
4002                            <1> _pregw2:
4003 00014DE9 89D8              <1>         mov     eax, ebx        ; restore eax
4004 00014DEB 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
4005 00014DF0 89CA              <1>         mov     edx, ecx        ; restore dx
4006 00014DF2 59                <1>       pop   ecx
4007 00014DF3 5B                <1>       pop   ebx
4008 00014DF4 C3                <1>       retn
4009                            <1>
4010                            <1> pciRegWrite8:
4011 00014DF5 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit write size
4012 00014DFA EBAD              <1>         jmp  short pciRegWrite   ; call generic PCI access
4013                            <1>
4014                            <1> pciRegWrite16:
4015 00014DFC 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit write size
4016 00014E01 0D00000040        <1>         or      eax, PCI16             ; call generic PCI access
4017 00014E06 EBA1              <1>         jmp  short pciRegWrite
4018                            <1>
4019                            <1> pciRegWrite32:
4020 00014E08 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit write size
4021 00014E0D 0D00000080        <1>         or      eax, PCI32             ; call generic PCI access
4022 00014E12 EB95              <1>         jmp  pciRegWrite
4023                            <1>
4024                            <1> init_codec:
4025                            <1>       ; 05/06/2017
4026                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4027                            <1>       ;
4028 00014E14 A1[789B0100]      <1>       mov   eax, [audio_dev_id]
4029 00014E19 B041              <1>       mov   al, VIA_ACLINK_CTRL
4030 00014E1B E86AFFFFFF        <1>       call  pciRegRead8
```

```
4031                                    <1>       ; ?
4032 00014E20 B040                      <1>       mov    al, VIA_ACLINK_STAT
4033 00014E22 E863FFFFFF                <1>       call   pciRegRead8
4034 00014E27 F6C201                    <1>       test   dl, VIA_ACLINK_C00_READY
4035 00014E2A 7508                      <1>         jnz     short _codec_ready_1
4036 00014E2C E80E000000                <1>       call   reset_codec
4037 00014E31 7306                      <1>       jnc    short _codec_ready_2 ; eax = 1
4038 00014E33 C3                        <1>       retn
4039                                    <1> _codec_ready_1:
4040 00014E34 B801000000                <1>       mov    eax, 1
4041                                    <1> _codec_ready_2:
4042 00014E39 E886000000                <1>       call   codec_io_w16
4043                                    <1> detect_codec:
4044 00014E3E C3                        <1>       retn
4045                                    <1>
4046                                    <1> reset_codec:
4047                                    <1>       ; 16/04/2017
4048                                    <1>       ; 23/03/2017
4049                                    <1>       ; ('codec.asm')
4050                                    <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4051 00014E3F A1[789B0100]              <1>       mov    eax, [audio_dev_id]
4052 00014E44 B041                      <1>       mov    al, VIA_ACLINK_CTRL
4053 00014E46 B2E0                      <1>              mov    dl, VIA_ACLINK_CTRL_ENABLE + VIA_ACLINK_CTRL_RESET + VIA_ACLINK_CTRL_SYNC
4054 00014E48 E8A8FFFFFF                <1>       call   pciRegWrite8
4055                                    <1>
4056 00014E4D E849000000                <1>       call   delay_100ms  ; wait 100 ms
4057                                    <1> _rc_cold:
4058 00014E52 E814000000                <1>         call    cold_reset
4059 00014E57 7301                      <1>         jnc     short _reset_codec_ok
4060                                    <1>
4061                                    <1>       ; 16/04/2017
4062                                    <1>         ;xor     eax, eax         ; timeout error
4063                                    <1>             ;stc
4064 00014E59 C3                        <1>       retn
4065                                    <1>
4066                                    <1> _reset_codec_ok:
4067                                    <1>       ; 01/09/2020
4068                                    <1>       ; 15/08/2020
4069                                    <1>       ; 27/07/2020
4070                                    <1>       ; also reset codec by using index control register 0 of AD1980 or ALC655
4071                                    <1>       ; (to fix line out -2 channels audio playing- problem on AD1980 codec)
4072                                    <1>
4073 00014E5A 29C0                      <1>       sub    eax, eax
4074 00014E5C BA00000000                <1>       mov    edx, CODEC_RESET_REG ; 00h ; Reset register
4075 00014E61 E8CA000000                <1>       call   codec_write
4076                                    <1>
4077                                    <1>       ;sub   eax, eax
4078                                    <1>       ; 01/09/2020
4079                                    <1>       ; 15/08/2020
4080                                    <1>       ; AD1980 BugFix
4081                                    <1>       ; (set HPSEL -headphone amp to be driven from mixer- and
4082                                    <1>       ;      CLDIS - center and LFE disable- bits)
4083                                    <1>       ;mov   eax, 0C00h ; HPSEL = bit 10, CLDIS = bit 11 ; 01/09/2020
4084                                    <1>       ;mov   edx, CODEC_MISC_CRTL_BITS_REG ; 76h ; Misc Ctrl Bits ; AD1980
4085                                    <1>       ;call  codec_write
4086                                    <1>
4087 00014E66 31C0                      <1>         xor     eax, eax
4088                                    <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
4089 00014E68 FEC0                      <1>         inc  al
4090 00014E6A C3                        <1>       retn
4091                                    <1>
4092                                    <1> cold_reset:
4093                                    <1>       ; 16/04/2017
4094                                    <1>       ; 23/03/2017
4095                                    <1>       ; ('codec.asm')
4096                                    <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4097                                    <1>       ;mov   eax, [audio_dev_id]
4098                                    <1>       ;mov   al, VIA_ACLINK_CTRL
4099 00014E6B 30D2                      <1>       xor    dl, dl ; 0
4100 00014E6D E883FFFFFF                <1>       call   pciRegWrite8
4101                                    <1>
4102 00014E72 E824000000                <1>       call   delay_100ms  ; wait 100 ms
4103                                    <1>
4104                                    <1>       ;; ACLink on, deassert ACLink reset, VSR, SGD data out
4105                                    <1>         ;; note - FM data out has trouble with non VRA codecs !!
4106                                    <1>
4107                                    <1>       ;mov   eax, [audio_dev_id]
4108                                    <1>       ;mov   al, VIA_ACLINK_CTRL
4109 00014E77 B2CC                      <1>       mov    dl, VIA_ACLINK_CTRL_INIT
4110 00014E79 E877FFFFFF                <1>       call   pciRegWrite8
4111                                    <1>
4112 00014E7E B910000000                <1>       mov    ecx, 16      ; total 2s
4113                                    <1>
4114                                    <1> _crst_wait:
4115                                    <1>       ;mov   eax, [audio_dev_id]
4116 00014E83 B040                      <1>       mov    al, VIA_ACLINK_STAT
4117 00014E85 E800FFFFFF                <1>       call   pciRegRead8
4118                                    <1>
4119 00014E8A F6C201                    <1>         test    dl, VIA_ACLINK_C00_READY
4120 00014E8D 750B                      <1>         jnz     short _crst_ok
4121                                    <1>
4122 00014E8F 51                        <1>       push   ecx
4123 00014E90 E806000000                <1>       call   delay_100ms
4124 00014E95 59                        <1>       pop    ecx
4125                                    <1>
4126 00014E96 49                        <1>         dec     ecx
4127 00014E97 75EA                      <1>         jnz     short _crst_wait
4128                                    <1>
4129                                    <1> _crst_fail:
4130 00014E99 F9                        <1>         stc
4131                                    <1> _crst_ok:
4132 00014E9A C3                        <1>       retn
4133                                    <1>
4134                                    <1> delay_100ms:
4135                                    <1>       ; 29/05/2017
```

```
4136                            <1>       ; 24/03/2017 ('codec.asm')
4137                            <1>       ; wait 100 ms
4138 00014E9B B990010000        <1>       mov    ecx, 400  ; 400*0.25ms
4139                            <1> _delay_x_ms:
4140 00014EA0 E803000000        <1>       call   delay1_4ms
4141 00014EA5 E2F9              <1>         loop _delay_x_ms
4142 00014EA7 C3                <1>       retn
4143                            <1>
4144                            <1> ;       delay1_4ms - Delay for 1/4 millisecond.
4145                            <1> ;         1mS = 1000us
4146                            <1> ;       Entry:
4147                            <1> ;         None
4148                            <1> ;       Exit:
4149                            <1> ;       None
4150                            <1> ;
4151                            <1> ;       Modified:
4152                            <1> ;         None
4153                            <1> ;
4154                            <1>
4155                            <1>       ; 29/05/2017
4156                            <1>       ; 23/04/2017
4157                            <1>       ; 05/03/2017 (TRDOS 386)
4158                            <1>       ; ('UTILS.ASM')
4159                            <1> delay1_4ms:
4160 00014EA8 50                <1>         push    eax
4161 00014EA9 51                <1>         push    ecx
4162 00014EAA B110              <1>         mov  cl, 16       ; close enough.
4163                            <1>
4164 00014EAC E461              <1>       in     al, PORTB ; 61h
4165                            <1>
4166 00014EAE 2410              <1>       and    al, REFRESH_STATUS ; 10h
4167 00014EB0 88C5              <1>       mov    ch, al       ; Start toggle state
4168                            <1>  d4ms1:
4169 00014EB2 E461              <1>       in     al, PORTB    ; Read system control port
4170                            <1>
4171 00014EB4 2410              <1>       and    al, REFRESH_STATUS ; Refresh toggles 15.085 microseconds
4172 00014EB6 38C5              <1>       cmp    ch, al
4173 00014EB8 74F8              <1>       je     short _d4ms1 ; Wait for state change
4174                            <1>
4175 00014EBA 88C5              <1>       mov    ch, al       ; Update with new state
4176 00014EBC FEC9              <1>       dec    cl
4177 00014EBE 75F2              <1>       jnz    short _d4ms1
4178                            <1>
4179 00014EC0 F8                <1>       clc    ; 29/05/2017
4180                            <1>
4181 00014EC1 59                <1>         pop     ecx
4182 00014EC2 58                <1>         pop     eax
4183 00014EC3 C3                <1>         retn
4184                            <1>
4185                            <1> ; 10/04/2017 (TRDOS 386)
4186                            <1> ; 12/11/2016
4187                            <1>
4188                            <1> codec_io_w16: ;w32
4189                            <1>       ; ('codec.asm')
4190 00014EC4 668B15[769B0100]  <1>         mov  dx, [audio_io_base]
4191 00014ECB 6681C28000        <1>         add     dx, VIA_REG_AC97
4192 00014ED0 EF                <1>       out    dx, eax
4193 00014ED1 C3                <1>         retn
4194                            <1>
4195                            <1> codec_io_r16: ;r32
4196                            <1>       ; ('codec.asm')
4197 00014ED2 668B15[769B0100]  <1>         mov     dx, [audio_io_base]
4198 00014ED9 6681C28000        <1>         add     dx, VIA_REG_AC97
4199 00014EDE ED                <1>         in   eax, dx
4200 00014EDF C3                <1>         retn
4201                            <1>
4202                            <1> ctrl_io_w8:
4203                            <1>       ; ('codec.asm')
4204 00014EE0 660315[769B0100]  <1>         add     dx, [audio_io_base]
4205 00014EE7 EE                <1>         out  dx, al
4206 00014EE8 C3                <1>         retn
4207                            <1>
4208                            <1> ctrl_io_r8:
4209                            <1>       ; ('codec.asm')
4210 00014EE9 660315[769B0100]  <1>         add     dx, [audio_io_base]
4211 00014EF0 EC                <1>         in   al, dx
4212 00014EF1 C3                <1>         retn
4213                            <1>
4214                            <1> ctrl_io_w32:
4215                            <1>       ; ('codec.asm')
4216 00014EF2 660315[769B0100]  <1>         add     dx, [audio_io_base]
4217 00014EF9 EF                <1>         out  dx, eax
4218 00014EFA C3                <1>         retn
4219                            <1>
4220                            <1> ctrl_io_r32:
4221                            <1>       ; ('codec.asm')
4222 00014EFB 660315[769B0100]  <1>         add  dx, [audio_io_base]
4223 00014F02 ED                <1>       in     eax, dx
4224 00014F03 C3                <1>         retn
4225                            <1>
4226                            <1> codec_read:
4227                            <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4228                            <1>         ; Use only primary codec.
4229                            <1>         ; eax = register
4230 00014F04 C1E010            <1>         shl     eax, VIA_REG_AC97_CMD_SHIFT
4231 00014F07 0D00008002        <1>         or      eax, VIA_REG_AC97_PRIMARY_VALID + VIA_REG_AC97_READ
4232                            <1>
4233 00014F0C E8B3FFFFFF        <1>       call    codec_io_w16
4234                            <1>
4235                            <1>             ; codec_valid
4236 00014F11 E831000000        <1>       call   codec_check_ready
4237 00014F16 7301              <1>         jnc  short _cr_ok
4238                            <1>
4239 00014F18 C3                <1>       retn
4240                            <1>
```

```
4241                            <1> _cr_ok:
4242                            <1>       ; wait 25 ms
4243 00014F19 B950000000        <1>       mov    ecx, 80 ; (100*0.25 ms)
4244                            <1> _cr_wloop:
4245 00014F1E E885FFFFFF        <1>       call   delay1_4ms
4246 00014F23 E2F9              <1>       loop   _cr_wloop
4247                            <1>
4248 00014F25 E8A8FFFFFF        <1>         call    codec_io_r16
4249 00014F2A 25FFFF0000        <1>         and     eax, 0FFFFh
4250 00014F2F C3                <1>         retn
4251                            <1>
4252                            <1> codec_write:
4253                            <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4254                            <1>         ; Use only primary codec.
4255                            <1>
4256                            <1>       ; eax = data (volume)
4257                            <1>       ; edx = register (mixer register)
4258                            <1>
4259 00014F30 C1E210            <1>       shl     edx, VIA_REG_AC97_CMD_SHIFT
4260                            <1>
4261 00014F33 C1E000            <1>         shl     eax, VIA_REG_AC97_DATA_SHIFT ; shl eax, 0
4262 00014F36 09C2              <1>         or      edx, eax
4263                            <1>
4264 00014F38 B800000000        <1>         mov     eax, VIA_REG_AC97_CODEC_ID_PRIMARY
4265 00014F3D C1E01E            <1>         shl     eax, VIA_REG_AC97_CODEC_ID_SHIFT
4266 00014F40 09D0              <1>         or      eax, edx
4267                            <1>
4268 00014F42 E87DFFFFFF        <1>         call    codec_io_w16
4269                            <1>         ;mov    [codec.regs+esi], ax
4270                            <1>
4271                            <1>         ;call      codec_check_ready
4272                            <1>             ;retn
4273                            <1>       ;jmp   short _codec_check_ready
4274                            <1>
4275                            <1> codec_check_ready:
4276                            <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4277                            <1>
4278                            <1> _codec_check_ready:
4279 00014F47 B914000000        <1>       mov    ecx, 20      ; total 2s
4280                            <1> _ccr_wait:
4281 00014F4C 51                <1>       push   ecx
4282                            <1>
4283 00014F4D E880FFFFFF        <1>         call    codec_io_r16
4284 00014F52 A900000001        <1>         test    eax, VIA_REG_AC97_BUSY
4285 00014F57 740B              <1>         jz      short _ccr_ok
4286                            <1>
4287 00014F59 E83DFFFFFF        <1>       call   delay_100ms
4288                            <1>
4289 00014F5E 59                <1>       pop    ecx
4290                            <1>
4291 00014F5F 49                <1>       dec     ecx
4292 00014F60 75EA              <1>         jnz     short _ccr_wait
4293                            <1>
4294 00014F62 F9                <1>         stc
4295 00014F63 C3                <1>         retn
4296                            <1>
4297                            <1> _ccr_ok:
4298 00014F64 59                <1>       pop    ecx
4299 00014F65 25FFFF0000        <1>       and     eax, 0FFFFh
4300 00014F6A C3                <1>         retn
4301                            <1>
4302                            <1> codec_config:
4303                            <1>       ; 10/06/2017
4304                            <1>       ; 29/05/2017
4305                            <1>       ; 24/04/2017
4306                            <1>       ; 21/04/2017
4307                            <1>       ; 16/04/2017 (TRDOS 386 Kernel)
4308                            <1>       ; 15/11/2016 ('codec.asm', 'player.com')
4309                            <1>       ; 14/11/2016
4310                            <1>       ; 12/11/2016 - Erdogan Tan
4311                            <1>       ;           (Ref: KolibriOS, 'setup_codec', codec.inc)
4312                            <1>
4313 00014F6B B802020000        <1>       mov     eax, 0202h
4314 00014F70 66A3[A69B0100]    <1>       mov    [audio_master_volume], ax
4315 00014F76 66B81F1F          <1>       mov    ax, 1F1Fh ; 31,31
4316 00014F7A BA02000000        <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
4317 00014F7F E8ACFFFFFF        <1>       call   codec_write
4318                            <1>       ;jc    short cconfig_error
4319                            <1>
4320                            <1>       ;mov    eax, 0202h
4321 00014F84 66B80202          <1>       mov     ax, 0202h
4322 00014F88 BA18000000        <1>       mov    edx, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
4323 00014F8D E89EFFFFFF        <1>       call   codec_write
4324                            <1>       ;jc    short cconfig_error
4325                            <1>
4326                            <1>       ;mov    eax, 0202h
4327 00014F92 66B80202          <1>       mov    ax, 0202h
4328 00014F96 BA04000000        <1>       mov    edx, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
4329 00014F9B E890FFFFFF        <1>       call   codec_write
4330                            <1>       ;jc    short cconfig_error
4331                            <1>
4332                            <1>       ;mov    eax, 08h
4333                            <1>         ;mov    ax,  08h
4334 00014FA0 66B80880          <1>       mov    ax, 8008h ; Mute
4335 00014FA4 BA0C000000        <1>       mov    edx, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
4336 00014FA9 E882FFFFFF        <1>       call   codec_write
4337                            <1>       ;jc    short cconfig_error
4338                            <1>
4339                            <1>       ;mov    eax, 0808h
4340 00014FAE 66B80808          <1>         mov     ax,  0808h
4341 00014FB2 BA10000000        <1>         mov  edx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
4342 00014FB7 E874FFFFFF        <1>       call   codec_write
4343                            <1>       ;jc    short cconfig_error
4344                            <1>
4345                            <1>       ;mov    eax, 0808h
```

```
4346 00014FBC 66B80808           <1>       mov     ax,  0808h
4347 00014FC0 BA12000000         <1>         mov  edx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
4348 00014FC5 E866FFFFFF         <1>       call   codec_write
4349                             <1>       ;jc    short cconfig_error
4350                             <1>
4351                             <1>       ;mov    eax, 0808h
4352 00014FCA 66B80808           <1>       mov     ax,  0808h
4353 00014FCE BA16000000         <1>         mov  edx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
4354                             <1>       ;call  codec_write
4355                             <1>       ;;jc   short cconfig_error
4356 00014FD3 E958FFFFFF         <1>       jmp    codec_write ; 10/06/2017
4357                             <1>
4358                             <1> ;     ; Extended Audio Status (2Ah)
4359                             <1> ;     mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
4360                             <1> ;     call   codec_read
4361                             <1> ;     and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
4362                             <1> ;     ;or     eax, 1                   ; set VRA (BIT0)
4363                             <1> ;     or     eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
4364                             <1> ;     mov    edx, CODEC_EXT_AUDIO_CTRL_REG
4365                             <1> ;     call   codec_write
4366                             <1> ;     ;jc    short cconfig_error
4367                             <1> ;
4368                             <1> ;set_sample_rate:
4369                             <1> ;     ;movzx eax, word [audio_freq]
4370                             <1> ;     mov    ax, [audio_freq]
4371                             <1> ;     mov    edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
4372                             <1> ;     ;call  codec_write
4373                             <1> ;     ;retn
4374                             <1> ;     jmp    codec_write
4375                             <1>
4376                             <1> ;cconfig_error:
4377                             <1> ;     retn
4378                             <1>
4379                             <1> vt8233_int_handler:
4380                             <1>       ; 27/07/2020
4381                             <1>       ; 22/07/2020
4382                             <1>       ; Interrupt Handler for VIA VT8237R Audio Controller
4383                             <1>       ; Note: called by 'dev_IRQ_service'
4384                             <1>       ; 14/10/2017
4385                             <1>       ; 09/10/2017, 10/10/2017, 12/10/2017
4386                             <1>       ; 13/06/2017
4387                             <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
4388                             <1>       ; 24/03/2017 - 'PLAYER.COM' ('player.asm')
4389                             <1>
4390                             <1>       ;push  eax ; * must be saved !
4391                             <1>       ;push  edx
4392                             <1>       ;push  ecx
4393                             <1>       ;push  ebx ; * must be saved !
4394                             <1>       ;push  esi
4395                             <1>       ;push  edi
4396                             <1>
4397                             <1>       ;cmp   byte [audio_busy], 1
4398                             <1>       ;jnb   short _ih0 ; 09/10/2017
4399                             <1>
4400                             <1>       ;mov   byte [audio_flag_eol], 0
4401                             <1>
4402 00014FD8 66BA0000           <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
4403 00014FDC E808FFFFFF         <1>         call    ctrl_io_r8
4404                             <1>
4405 00014FE1 A880               <1>       test    al, VIA_REG_STAT_ACTIVE
4406 00014FE3 7417               <1>         jz      short _ih0 ; 09/10/2017
4407                             <1>
4408 00014FE5 2407               <1>         and     al, VIA_REG_STAT_EOL + VIA_REG_STAT_FLAG + VIA_REG_STAT_STOPPED
4409 00014FE7 A2[A59B0100]       <1>       mov    [audio_flag_eol], al
4410 00014FEC 740E               <1>         jz   short _ih0 ; 09/10/2017
4411                             <1>
4412                             <1>       ; 09/10/2017
4413                             <1>       ;mov   byte [audio_busy], 1
4414                             <1>
4415 00014FEE 803D[A49B0100]01   <1>       cmp    byte [audio_play_cmd], 1
4416 00014FF5 7315               <1>       jnb    short _ih1 ; 10/10/2017
4417                             <1>
4418 00014FF7 E84A000000         <1>       call   channel_reset
4419                             <1> _ih0:
4420                             <1>       ; 09/10/2017
4421 00014FFC A0[A59B0100]       <1>         mov     al, [audio_flag_eol]   ;; ack ;;
4422 00015001 66BA0000           <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
4423 00015005 E8D6FEFFFF         <1>         call    ctrl_io_w8
4424 0001500A EB39               <1>       jmp    short _ih4
4425                             <1> _ih1:
4426                             <1> vt8233_tuneLoop:
4427 0001500C A0[A59B0100]       <1>         mov     al, [audio_flag_eol]   ;; ack ;;
4428 00015011 66BA0000           <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
4429 00015015 E8C6FEFFFF         <1>         call    ctrl_io_w8
4430                             <1>
4431                             <1>       ; 22/07/2020
4432                             <1>       ;; 12/10/2017
4433                             <1>       ;mov   byte [audio_flag], 0 ; Reset
4434                             <1>
4435                             <1>       ; 10/10/2017
4436                             <1>       ; 09/10/2017
4437                             <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_FLAG
4438                             <1>       ;jz    short _ih2 ; EOL
4439                             <1>
4440                             <1>       ; 22/07/2020
4441                             <1>       ; 14/10/2017
4442                             <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_EOL
4443                             <1>       ;jnz   short _ih2 ; EOL
4444                             <1>       ;               ; (Half Buffer 2 has been completed
4445                             <1>       ;               ; and Half Buffer 1 will be played.)
4446                             <1>
4447                             <1>       ; FLAG
4448                             <1>       ; (Half Buffer 1 has been completed
4449                             <1>       ;  and Half Buffer 2 will be played.)
4450                             <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
0001501A 8B3D[909B0100]         <1>       mov    edi, [audio_dma_buff]
00015020 8B0D[949B0100]         <1>       mov    ecx, [audio_dmabuff_size]
00015026 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
00015028 F605[989B0100]01       <1>       test   byte [audio_flag], 1  ; Current flag value
0001502F 7402                   <1>       jz     short _ih3 ; Half Buffer 1 must be filled
                                <1>
                                <1>       ; Half Buffer 2 must be filled
00015031 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>
00015033 8B35[889B0100]         <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
00015039 C1E902                 <1>       shr    ecx, 2 ; half buff size / 4
0001503C F3A5                   <1>       rep    movsd
                                <1>
                                <1>       ; switch flag value ;
0001503E 8035[989B0100]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>
00015045 C3                     <1>       retn
                                <1>
                                <1> channel_reset:
                                <1>       ; 24/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 23/03/2017
                                <1>       ; 14/11/2016 - Erdogan Tan
                                <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
00015046 BA01000000             <1>         mov  edx, VIA_REG_OFFSET_CONTROL
                                <1>         ;mov eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE + VIA_REG_CTRL_RESET
0001504B B848000000             <1>         mov  eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE ; 24/06/2017
00015050 E88BFEFFFF             <1>       call    ctrl_io_w8
                                <1>
                                <1>         ;mov edx, VIA_REG_OFFSET_CONTROL
                                <1>         ;call   ctrl_io_r8
                                <1>
                                <1>       ; wait for 50 ms
00015055 B9A0000000             <1>       mov    ecx, 160 ; (200*0.25 ms) ; 29/05/2017
                                <1> _ch_rst_wait:
0001505A E849FEFFFF             <1>       call   delay1_4ms
0001505F 49                     <1>       dec    ecx
00015060 75F8                   <1>       jnz    short _ch_rst_wait
                                <1>
                                <1>         ; disable interrupts
00015062 BA01000000             <1>         mov  edx, VIA_REG_OFFSET_CONTROL
00015067 31C0                   <1>         xor     eax, eax
00015069 E872FEFFFF             <1>         call    ctrl_io_w8
                                <1>
                                <1>         ; clear interrupts
0001506E BA00000000             <1>         mov  edx, VIA_REG_OFFSET_STATUS
00015073 B803000000             <1>       mov    eax, 3
00015078 E863FEFFFF             <1>         call ctrl_io_w8
                                <1>
                                <1>       ;mov   edx, VIA_REG_OFFSET_CURR_PTR
                                <1>       ;xor   eax, eax
                                <1>       ;call  ctrl_io_w32
                                <1>
0001507D C3                     <1>         retn
                                <1>
                                <1> vt8233_stop: ; 22/04/2017
0001507E C605[A49B0100]00       <1>       mov    byte [audio_play_cmd], 0 ; stop !
                                <1> _tlp2:
                                <1>       ; 24/06/2017
                                <1>         ; finished with song, stop everything
                                <1>       ;mov   al, VIA_REG_CTRL_INT
                                <1>         ;or  al, VIA_REG_CTRL_TERMINATE
                                <1>       ;mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
                                <1>         ;call      ctrl_io_w8
                                <1>
                                <1>         ;call      channel_reset
```

```
4556                                    <1>       ;retn
4557                                    <1>
4558 00015085 EBBF                      <1>       jmp    short channel_reset
4559                                    <1>
4560                                    <1> set_vt8233_bdl: ; Set VT8237R Buffer Descriptor List
4561                                    <1>       ; 22/07/2020 - TRDOS 386 v2.0.2
4562                                    <1>       ; 28/05/2017
4563                                    <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
4564                                    <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
4565                                    <1>
4566                                    <1>       ; eax = dma buffer address = [audio_DMA_buff]
4567                                    <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
4568                                    <1>
4569 00015087 D1E9                      <1>       shr    ecx, 1 ; dma half buffer size
4570 00015089 89CE                      <1>       mov    esi, ecx
4571                                    <1>
4572 0001508B BF[AC9B0100]              <1>         mov     edi, audio_bdl_buff    ; get BDL address
4573 00015090 B910000000                <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
4574                                    <1>
4575 00015095 EB05                      <1>       jmp    short s_vt8233_bdl1
4576                                    <1>
4577                                    <1> s_vt8233_bdl0:
4578                                    <1>       ; set buffer descriptor 0 to start of data file in memory
4579                                    <1>
4580 00015097 A1[909B0100]              <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
4581                                    <1>
4582                                    <1> s_vt8233_bdl1:
4583 0001509C AB                        <1>       stosd                        ; store dmabuffer1 address
4584                                    <1>
4585 0001509D 89C2                      <1>       mov    edx, eax
4586                                    <1>
4587                                    <1> ; VIA VT8235.PDF: (Page 110) (Erdogan Tan, 29/11/2016)
4588                                    <1>       ;
4589                                    <1>       ;      Audio SGD Table Format
4590                                    <1>       ;      ------------------------------
4591                                    <1>       ;      63   62    61-56    55-32  31-0
4592                                    <1>       ;      --   --   --------  -----  ----
4593                                    <1>       ;      EOL FLAG -reserved- Base   Base
4594                                    <1>       ;                          Count  Address
4595                                    <1>       ;                           [23:0] [31:0]
4596                                    <1>       ;      EOL: End Of Link.
4597                                    <1>       ;           1 indicates this block is the last of the link.
4598                                    <1>       ;           If the channel "Interrupt on EOL" bit is set, then
4599                                    <1>       ;           an interrupt is generated at the end of the transfer.
4600                                    <1>       ;
4601                                    <1>       ;      FLAG: Block Flag. If set, transfer pauses at the end of this
4602                                    <1>       ;            block. If the channel "Interrupt on FLAG" bit is set,
4603                                    <1>       ;            then an interrupt is generated at the end of this block.
4604                                    <1>
4605 0001509F 89F0                      <1>      mov    eax, esi ; DMA half buffer size
4606 000150A1 01C2                      <1>      add    edx, eax
4607 000150A3 0D00000040                <1>      or     eax, FLAG
4608                                    <1>      ;or    eax, EOL
4609 000150A8 AB                        <1>      stosd
4610                                    <1>
4611                                    <1> ; 2nd buffer:
4612                                    <1>
4613 000150A9 89D0                      <1>        mov  eax, edx ; Physical address of the 2nd half of DMA buffer
4614 000150AB AB                        <1>      stosd         ; store dmabuffer2 address
4615                                    <1>
4616                                    <1> ; set length to [audio_dmabuff_size]/2
4617                                    <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
4618                                    <1> ;
4619 000150AC 89F0                      <1>      mov    eax, esi ; DMA half buffer size
4620                                    <1>      ; 22/07/2020
4621                                    <1>      ;or    eax, EOL
4622 000150AE 0D00000040                <1>      or     eax, FLAG
4623 000150B3 AB                        <1>      stosd
4624                                    <1>
4625 000150B4 E2E1                      <1>      loop    s_vt8233_bdl0
4626                                    <1>
4627                                    <1>      ; 22/07/2020
4628 000150B6 814FFC00000080            <1>      or     dword [edi-4], EOL
4629                                    <1>
4630 000150BD C3                        <1>      retn
4631                                    <1>
4632                                    <1> vt8233_start_play:
4633                                    <1>      ; 01/09/2020
4634                                    <1>      ; 22/07/2020
4635                                    <1>      ; start to play audio data via VT8233 audio controller
4636                                    <1>      ; 13/06/2017
4637                                    <1>      ; 10/06/2017
4638                                    <1>      ; 24/04/2017
4639                                    <1>      ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
4640                                    <1>      ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
4641                                    <1>      ; write buffer descriptor list address
4642                                    <1>
4643                                    <1>      ; Extended Audio Status (2Ah)
4644 000150BE B82A000000                <1>      mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
4645 000150C3 E83CFEFFFF                <1>      call   codec_read
4646 000150C8 25FDFF0000                <1>      and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
4647                                    <1>      ;or     eax, 1                   ; set VRA (BIT0)
4648                                    <1>      ;or    eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
4649 000150CD 0C05                      <1>      or     al, 5
4650                                    <1>      ; 01/09/2020
4651                                    <1>      ;or    eax, 3805h ; AD1980 (PRK, PRJ, PRI = 1 .. only front DAC)
4652                                    <1>      ; 01/09/2020
4653                                    <1>      ;mov   edx, CODEC_EXT_AUDIO_CTRL_REG
4654                                    <1>      ;cmp   word [audio_freq], 0BB80h ; 48 kHz
4655                                    <1>      ;jne   short set_extd_audio_status_1
4656                                    <1>      ;and   al, 0FEh ; disable VRA bit (set sample rate to 48000 Hz)
4657                                    <1>      ;jmp   short set_extd_audio_status_2
4658                                    <1> ;set_extd_audio_status_1:
4659 000150CF BA2A000000                <1>      mov    edx, CODEC_EXT_AUDIO_CTRL_REG
4660 000150D4 E857FEFFFF                <1>      call   codec_write
```

```
                                <1>       ;jc    short cconfig_error
                                <1>
                                <1> set_sample_rate:
                                <1>       ;movzx eax, word [audio_freq]
000150D9 66A1[A29B0100]         <1>       mov    ax, [audio_freq]
000150DF BA2C000000             <1>       mov    edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
                                <1> ;set_extd_audio_status_2:
000150E4 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>
000150E9 B8[AC9B0100]           <1>         mov  eax, audio_bdl_buff
                                <1>
                                <1>       ; 12/11/2016 - Erdogan Tan
                                <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
000150EE BA04000000             <1>       mov    edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
000150F3 E8FAFDFFFF             <1>         call ctrl_io_w32
                                <1>
                                <1>       ;call  codec_check_ready
                                <1>
000150F8 66BA0200               <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_L
                                <1>         ;mov eax, 2 ; 31
000150FC B01F                   <1>       mov    al, 31
000150FE 2A05[A69B0100]         <1>         sub  al, [audio_master_volume_l]
00015104 E8D7FDFFFF             <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
0001511A B0FF                   <1>       mov    al, 0FFh
0001511C E813000000             <1>       call    set_VT8233_LastValidIndex
                                <1>
00015121 C605[A49B0100]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
00015128 B0A1                   <1>       mov    al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START + VIA_REG_CTRL_INT_FLAG
                                <1>
0001512A 66BA0100               <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
0001512E E8ADFDFFFF             <1>         call    ctrl_io_w8
                                <1>       ;call  codec_check_ready
                                <1>       ;retn
                                <1>       ;jmp   codec_check_ready
00015133 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
00015134 50                     <1>       push   eax ; 23/07/2020
                                <1>       ;push  ecx
00015135 0FB705[A29B0100]       <1>       movzx  eax, word [audio_freq] ; Hertz
0001513C BA00001000             <1>       mov    edx, 100000h ; 2^20 = 1048576
00015141 F7E2                   <1>       mul    edx
00015143 B980BB0000             <1>       mov    ecx, 48000
00015148 F7F1                   <1>       div    ecx
                                <1>       ;and   eax, 0FFFFFh
                                <1>       ;pop   ecx
                                <1>       ;pop   dx
0001514A 5A                     <1>       pop    edx ; 23/07/2020
0001514B C1E218                 <1>       shl    edx, 24  ; STOP Index Setting: Bit 24 to 31
```

```
4766 0001514E 09D0                <1>       or     eax, edx
4767                              <1>       ; 19/11/2016
4768 00015150 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16
4769 00015157 7505                <1>       jne    short sLVI_1
4770 00015159 0D00002000          <1>       or     eax, VIA8233_REG_TYPE_16BIT
4771                              <1> sLVI_1:
4772 0001515E 803D[A19B0100]02    <1>       cmp    byte [audio_stmo], 2
4773 00015165 7505                <1>       jne    short sLVI_2
4774 00015167 0D00001000          <1>       or     eax, VIA8233_REG_TYPE_STEREO
4775                              <1> sLVI_2:
4776 0001516C BA08000000          <1>       mov     edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
4777 00015171 E87CFDFFFF          <1>         call    ctrl_io_w32
4778                              <1>       ;call  codec_check_ready
4779                              <1>       ;pop   edx
4780 00015176 C3                  <1>       retn
4781                              <1>
4782                              <1> vt8233_pause: ; pause
4783                              <1>       ; 10/06/2017
4784                              <1>       ; 22/04/2017
4785                              <1>       ;mov     al, VIA_REG_CTRL_INT
4786                              <1>         ;or      al, VIA_REG_CTRL_PAUSE
4787                              <1>       ; 23/07/2020
4788 00015177 B029                <1>       mov    al, VIA_REG_CTRL_PAUSE+VIA_REG_CTRL_INT_FLAG+VIA_REG_CTRL_AUTOSTART
4789                              <1>
4790 00015179 66BA0100            <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
4791 0001517D E85EFDFFFF          <1>         call    ctrl_io_w8
4792                              <1>       ;call  codec_check_ready
4793                              <1>       ;retn
4794                              <1>       ;jmp   codec_check_ready
4795 00015182 C3                  <1>       retn
4796                              <1>
4797                              <1> vt8233_reset:
4798                              <1>       ; 22/04/2017
4799                              <1>       ; reset VT8237R (vt8233) Audio Controller
4800                              <1>       ;cmp   byte [audio_play_cmd], 1
4801                              <1>       ;jna   short vt8233_rst_0
4802 00015183 C605[A49B0100]00    <1>       mov    byte [audio_play_cmd], 0 ; stop !
4803                              <1> vt8233_rst_0:
4804 0001518A E8B0FCFFFF          <1>       call   reset_codec
4805 0001518F 720A                <1>       jc     short vt8233_rst_1 ; codec error !
4806                              <1>       ; eax = 1
4807 00015191 E82EFDFFFF          <1>       call   codec_io_w16 ; w32
4808 00015196 E8ABFEFFFF          <1>       call   channel_reset
4809                              <1> vt8233_rst_1:
4810 0001519B C3                  <1>       retn
4811                              <1>
4812                              <1> vt8233_volume:
4813                              <1>       ; set VT8237R (vt8233) sound volume level
4814                              <1>       ; 24/04/2017
4815                              <1>       ; 22/04/2017
4816                              <1>       ; bl = component (0 = master/playback/lineout volume)
4817                              <1>       ; cl = left channel volume level (0 to 31)
4818                              <1>       ; ch = right channel volume level (0 to 31)
4819                              <1>
4820 0001519C 08DB                <1>       or     bl, bl
4821 0001519E 7520                <1>       jnz    short vt8233_vol_1 ; temporary !
4822 000151A0 66B81F1F            <1>       mov    ax, 1F1Fh ; 31,31
4823 000151A4 38C1                <1>       cmp    cl, al
4824 000151A6 7718                <1>       ja     short vt8233_vol_1 ; temporary !
4825 000151A8 38E5                <1>       cmp    ch, ah
4826 000151AA 7714                <1>       ja     short vt8233_vol_1 ; temporary !
4827 000151AC 66890D[A69B0100]    <1>       mov    [audio_master_volume], cx
4828 000151B3 6629C8              <1>       sub    ax, cx
4829 000151B6 BA02000000          <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
4830 000151BB E870FDFFFF          <1>       call   codec_write
4831                              <1> vt8233_vol_1:
4832 000151C0 C3                  <1>       retn
4833                              <1>
4834                              <1> ; CODE for SOUND BLASTER 16
4835                              <1>
4836                              <1> DetectSB:
4837                              <1>       ; 24/04/2017
4838                              <1>       ;pushad
4839                              <1> ScanPort:
4840 000151C1 66BB1002            <1>       mov     bx, 210h    ; start scanning ports
4841                              <1>                           ; 210h, 220h, .. 260h
4842                              <1> ResetDSP:
4843 000151C5 6689DA              <1>       mov     dx, bx            ; try to reset the DSP.
4844 000151C8 6683C206            <1>       add     dx, 06h
4845 000151CC B001                <1>       mov    al, 1
4846 000151CE EE                  <1>       out    dx, al
4847                              <1>
4848 000151CF EC                  <1>       in     al, dx
4849 000151D0 EC                  <1>       in     al, dx
4850 000151D1 EC                  <1>       in     al, dx
4851 000151D2 EC                  <1>       in     al, dx
4852                              <1>
4853 000151D3 30C0                <1>       xor     al, al
4854 000151D5 EE                  <1>       out    dx, al
4855                              <1>
4856 000151D6 6683C208            <1>       add     dx, 08h
4857 000151DA 66B96400            <1>       mov    cx, 100
4858                              <1> WaitID:
4859 000151DE EC                  <1>       in     al, dx
4860 000151DF 08C0                <1>       or      al, al
4861 000151E1 7804                <1>       js      short GetID
4862 000151E3 E2F9                <1>       loop    WaitID
4863 000151E5 EB0F                <1>       jmp     short NextPort
4864                              <1> GetID:
4865 000151E7 6683EA04            <1>       sub     dx, 04h
4866 000151EB EC                  <1>       in     al, dx
4867 000151EC 3CAA                <1>       cmp     al, 0AAh
4868 000151EE 7413                <1>       je      short Found
4869 000151F0 6683C204            <1>       add     dx, 04h
4870 000151F4 E2E8                <1>       loop    WaitID
```

```
                                <1> NextPort:
000151F6 6683C310               <1>       add     bx, 10h           ; if not response,
000151FA 6681FB6002             <1>       cmp     bx, 260h    ; try the next port.
000151FF 76C4                   <1>       jbe     short ResetDSP
00015201 F9                     <1>       stc
00015202 C3                     <1>       retn
                                <1> Found:
00015203 66891D[769B0100]       <1>       mov     [audio_io_base], bx      ; SB Port Address Found!
                                <1> ScanIRQ:
                                <1> SetIrqs:
0001520A 28C0                   <1>       sub    al, al ; 0
0001520C A2[6E9B0100]           <1>       mov    [IRQnum], al ; reset
00015211 A2[739B0100]           <1>       mov    [audio_intr], al ; reset
                                <1>
                                <1>       ; ah > 0 -> set IRQ vector
                                <1>       ; al = IRQ number
                                <1>       ;mov   ax, 103h ; IRQ 3
                                <1>       ;call  set_hardware_int_vector
                                <1>       ;mov   ax, 104h ; IRQ 4
                                <1>       ;call  set_hardware_int_vector
00015216 66B80501               <1>       mov    ax, 105h ; IRQ 5
0001521A E8CADDFFFF             <1>       call   set_hardware_int_vector
0001521F 66B80701               <1>       mov    ax, 107h ; IRQ 7
00015223 E8C1DDFFFF             <1>       call   set_hardware_int_vector
                                <1>
00015228 668B15[769B0100]       <1>       mov     dx, [audio_io_base] ; tells to the SB to
0001522F 6683C20C               <1>       add     dx, 0Ch               ; generate a IRQ!
                                <1> WaitSb:
00015233 EC                     <1>       in     al, dx
00015234 08C0                   <1>       or      al, al
00015236 78FB                   <1>       js      short WaitSb
00015238 B0F2                   <1>       mov     al, 0F2h
0001523A EE                     <1>       out    dx, al
                                <1>
0001523B 31C9                   <1>       xor     ecx, ecx    ; wait until IRQ level
                                <1> WaitIRQ:
0001523D A0[6E9B0100]           <1>       mov    al, [IRQnum]
00015242 3C00                   <1>       cmp     al, 0 ; is changed or timeout.
00015244 7706                   <1>       ja     short IrqOk
00015246 6649                   <1>       dec    cx
00015248 75F3                   <1>       jnz    short WaitIRQ
0001524A EB15                   <1>       jmp    short RestoreIrqs
                                <1> IrqOk:
0001524C A2[739B0100]           <1>       mov    [audio_intr], al ; set
00015251 668B15[769B0100]       <1>       mov     dx, [audio_io_base]
00015258 6683C20E               <1>       add     dx, 0Eh
0001525C EC                     <1>       in     al, dx ; SB acknowledge.
0001525D B020                   <1>       mov    al, 20h
0001525F E620                   <1>       out    20h, al      ; Hardware acknowledge.
                                <1>
                                <1> RestoreIrqs:
                                <1>       ; ah = 0 -> reset IRQ vector
                                <1>       ; al = IRQ number
                                <1>       ;mov   ax, 3 ; IRQ 3
                                <1>       ;call  set_hardware_int_vector
                                <1>       ;mov   ax, 4 ; IRQ 4
                                <1>       ;call  set_hardware_int_vector
00015261 66B80500               <1>       mov    ax, 5 ; IRQ 5
00015265 E87FDDFFFF             <1>       call   set_hardware_int_vector
0001526A 66B80700               <1>       mov    ax, 7 ; IRQ 7
0001526E E876DDFFFF             <1>       call   set_hardware_int_vector
                                <1>
00015273 31D2                   <1>       xor    edx, edx
00015275 8915[789B0100]         <1>       mov    [audio_dev_id], edx ; 0
0001527B 8915[7C9B0100]         <1>       mov    [audio_vendor], edx ; 0
00015281 8915[809B0100]         <1>       mov    [audio_stats_cmd], edx ; 0
                                <1>
                                <1>       ;popad
                                <1>
00015287 803D[739B0100]01       <1>       cmp     byte [audio_intr], 1 ; IRQ level was changed?
                                <1>
0001528E C3                     <1>       retn
                                <1>
                                <1> %macro      SbOut  1
                                <1> %%Wait:
                                <1>       in     al, dx
                                <1>       or     al, al
                                <1>       js     short %%Wait
                                <1>       mov    al, %1
                                <1>       out    dx, al
                                <1> %endmacro
                                <1>
                                <1> SbInit_play:
                                <1>       ; 22/10/2017
                                <1>       ; 20/10/2017
                                <1>       ; 06/10/2017
                                <1>       ; 13/07/2017, 09/08/2017
                                <1>       ; 24/04/2017, 15/05/2017, 24/06/2017
                                <1>       ;pushad
                                <1> SetBuffer:
                                <1>       ;mov   byte [DmaFlag], 0
                                <1>
0001528F 8B1D[909B0100]         <1>       mov    ebx, [audio_dma_buff] ; physical addr of DMA buff
00015295 89DF                   <1>       mov    edi, ebx
00015297 8B0D[949B0100]         <1>       mov     ecx, [audio_dmabuff_size]
                                <1>
0001529D 803D[A09B0100]10       <1>       cmp    byte [audio_bps], 16
000152A4 7531                   <1>       jne    short sbInit_0 ; set 8 bit DMA buffer
                                <1>
                                <1>       ; 09/08/2017
                                <1>       ; convert byte count to word count
000152A6 D1E9                   <1>       shr    ecx, 1
000152A8 49                     <1>       dec    ecx ; word count - 1
                                <1>       ; convert byte offset to word offset
000152A9 D1EB                   <1>       shr    ebx, 1
```

```
                                <1>
                                <1>       ; 16 bit DMA buffer setting (DMA channel 5)
000152AB B005                   <1>       mov     al, 05h  ; set mask bit for channel 5  (4+1)
000152AD E6D4                   <1>       out    0D4h, al
                                <1>
000152AF 30C0                   <1>       xor     al, al   ; stops all DMA processes on selected channel
000152B1 E6D8                   <1>       out    0D8h, al ; clear selected channel register
                                <1>
000152B3 88D8                   <1>       mov     al, bl       ; byte 0 of DMA buffer offset in words (physical)
000152B5 E6C4                   <1>       out    0C4h, al ; DMA channel 5 port number
                                <1>
000152B7 88F8                   <1>       mov     al, bh   ; byte 1 of DMA buffer offset in words (physical)
000152B9 E6C4                   <1>       out    0C4h, al
                                <1>
                                <1>       ; 09/08/2017
000152BB C1EB0F                 <1>       shr    ebx, 15       ; complete 16 bit shift
000152BE 80E3FE                 <1>       and    bl, 0FEh ; clear bit 0 (not necessary, it will be ignored)
                                <1>
000152C1 88D8                   <1>       mov     al, bl   ; byte 2 of DMA buffer address (physical)
000152C3 E68B                   <1>       out    8Bh, al  ; page register port addr for channel 5 ; 13/07/2017
                                <1>
000152C5 88C8                   <1>       mov     al, cl   ; low byte of DMA count - 1
000152C7 E6C6                   <1>       out    0C6h, al ; count register port addr for channel 1
                                <1>
000152C9 88E8                   <1>       mov     al, ch   ; high byte of DMA count - 1
000152CB E6C6                   <1>       out    0C6h, al
                                <1>
                                <1>       ; channel 5, read, autoinitialized, single mode
                                <1>       ;mov   al, 49h
000152CD B059                   <1>       mov    al, 59h  ; 06/10/2017
000152CF E6D6                   <1>       out    0D6h, al ; DMA mode register port address
                                <1>
000152D1 B001                   <1>       mov     al, 01h  ; clear mask bit for channel 1
000152D3 E6D4                   <1>       out    0D4h, al ; DMA mask register port address
                                <1>
000152D5 EB28                   <1>       jmp    short ClearBuffer
                                <1>
                                <1> sbInit_0:
000152D7 49                     <1>       dec     ecx ; 09/08/2017
                                <1>
                                <1>       ; 8 bit DMA buffer setting (DMA channel 1)
000152D8 B005                   <1>       mov     al, 05h ; set mask bit for channel 1  (4+1)
000152DA E60A                   <1>       out    0Ah, al ; DMA mask register
                                <1>
000152DC 30C0                   <1>       xor     al, al  ; stops all DMA processes on selected channel
000152DE E60C                   <1>       out    0Ch, al ; clear selected channel register
                                <1>
000152E0 88D8                   <1>       mov     al, bl      ; byte 0 of DMA buffer address (physical)
000152E2 E602                   <1>       out    02h, al ; DMA channel 1 port number
                                <1>
000152E4 88F8                   <1>       mov     al, bh  ; byte 1 of DMA buffer address (physical)
000152E6 E602                   <1>       out    02h, al
                                <1>
000152E8 C1EB10                 <1>       shr    ebx, 16
                                <1>
000152EB 88D8                   <1>       mov     al, bl  ; byte 2 of DMA buffer address (physical)
000152ED E683                   <1>       out    83h, al ; page register port addr for channel 1
                                <1>
000152EF 88C8                   <1>       mov     al, cl  ; low byte of DMA count - 1
000152F1 E603                   <1>       out    03h, al ; count register port addr for channel 1
                                <1>
000152F3 88E8                   <1>       mov     al, ch  ; high byte of DMA count - 1
000152F5 E603                   <1>       out    03h, al
                                <1>
                                <1>       ; channel 1, read, autoinitialized, single mode
                                <1>       ;mov   al, 49h
000152F7 B059                   <1>       mov    al, 59h ; 06/10/2017
000152F9 E60B                   <1>       out    0Bh, al ; DMA mode register port address
                                <1>
000152FB B001                   <1>       mov     al, 01h ; clear mask bit for channel 1
000152FD E60A                   <1>       out    0Ah, al ; DMA mask register port address
                                <1>
                                <1> ClearBuffer:
                                <1>       ;;mov  edi, [audio_dma_buff]
                                <1>       ;;mov  ecx, [audio_dmabuff_size]
                                <1>       ;inc   ecx
                                <1>       ;mov     al, 80h
                                <1>       ;;cld
                                <1>       ;rep     stosb
                                <1> SetIrq:
                                <1>       ;mov   ebx, SbIrqhandler
                                <1>       ;mov   al, [audio_intr] ; IRQ number
                                <1>       ;call  set_dev_IRQ_service
                                <1>       ;; SETUP (audio) INTERRUPT CALLBACK SERVICE
                                <1>       ;mov   bl, [audio_intr] ; IRQ number
                                <1>       ;mov   bh, [audio_cb_mode]
                                <1>       ;inc   bh  ; 1 = Signal Response Byte method (fixed value)
                                <1>       ;          ; 2 = Callback service method
                                <1>       ;          ; 3 = Auto Increment S.R.B. method
                                <1>       ;mov   cl, [audio_srb]
                                <1>       ;mov   edx, [audio_cb_addr]
                                <1>       ;mov   al, [audio_user]
                                <1>       ;call  set_irq_callback_service
                                <1> ResetDsp:
000152FF 668B15[769B0100]       <1>       mov     dx, [audio_io_base]
00015306 6683C206               <1>       add     dx, 06h
0001530A B001                   <1>       mov     al, 1
0001530C EE                     <1>       out    dx, al
                                <1>
0001530D EC                     <1>       in     al, dx
0001530E EC                     <1>       in     al, dx
0001530F EC                     <1>       in     al, dx
00015310 EC                     <1>       in     al, dx
                                <1>
00015311 30C0                   <1>       xor     al, al
```

```
5081 00015313 EE                  <1>       out    dx, al
5082                              <1>
5083 00015314 66B96400            <1>       mov     cx, 100
5084 00015318 28E4                <1>       sub    ah, ah ; 0
5085                              <1> WaitId:
5086 0001531A 668B15[769B0100]    <1>       mov     dx, [audio_io_base]
5087 00015321 6683C20E            <1>       add     dx, 0Eh
5088 00015325 EC                  <1>       in     al, dx
5089 00015326 08C0                <1>       or      al, al
5090 00015328 7807                <1>       js      short sb_GetId
5091 0001532A E2EE                <1>       loop    WaitId
5092 0001532C E9B4000000          <1>       jmp     sb_Exit
5093                              <1> sb_GetId:
5094 00015331 668B15[769B0100]    <1>       mov     dx, [audio_io_base]
5095 00015338 6683C20A            <1>       add     dx, 0Ah
5096 0001533C EC                  <1>       in     al, dx
5097 0001533D 3CAA                <1>       cmp     al, 0AAh
5098 0001533F 7407                <1>       je      short SbOk
5099 00015341 E2D7                <1>       loop    WaitId
5100 00015343 E99D000000          <1>       jmp    sb_Exit
5101                              <1> SbOk:
5102 00015348 668B15[769B0100]    <1>       mov     dx, [audio_io_base]
5103 0001534F 6683C20C            <1>       add     dx, 0Ch
5104                              <1>       SbOut   0D1h ; Turn on speaker
4945                              <2> %%Wait:
4946 00015353 EC                  <2>  in al, dx
4947 00015354 08C0                <2>  or al, al
4948 00015356 78FB                <2>  js short %%Wait
4949 00015358 B0D1                <2>  mov al, %1
4950 0001535A EE                  <2>  out dx, al
5105                              <1>       SbOut   41h ; 8 bit or 16 bit transfer
4945                              <2> %%Wait:
4946 0001535B EC                  <2>  in al, dx
4947 0001535C 08C0                <2>  or al, al
4948 0001535E 78FB                <2>  js short %%Wait
4949 00015360 B041                <2>  mov al, %1
4950 00015362 EE                  <2>  out dx, al
5106 00015363 668B1D[A29B0100]    <1>       mov    bx, [audio_freq] ; sampling rate (Hz)
5107                              <1>       SbOut  bh ; sampling rate high byte
4945                              <2> %%Wait:
4946 0001536A EC                  <2>  in al, dx
4947 0001536B 08C0                <2>  or al, al
4948 0001536D 78FB                <2>  js short %%Wait
4949 0001536F 88F8                <2>  mov al, %1
4950 00015371 EE                  <2>  out dx, al
5108                              <1>       SbOut  bl ; sampling rate low byte
4945                              <2> %%Wait:
4946 00015372 EC                  <2>  in al, dx
4947 00015373 08C0                <2>  or al, al
4948 00015375 78FB                <2>  js short %%Wait
4949 00015377 88D8                <2>  mov al, %1
4950 00015379 EE                  <2>  out dx, al
5109                              <1>
5110                              <1>       ; 22/05/2017
5111 0001537A E8C0000000          <1>       call   sb16_volume_initial ; 15/05/2017
5112                              <1>       ; 20/05/2017
5113                              <1>       ;call  sb16_volume
5114                              <1>
5115                              <1> StartDma:
5116                              <1>       ; autoinitialized mode
5117 0001537F 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
5118 00015386 7411                <1>       je     short sb_play_1
5119                              <1>       ; 8 bit samples
5120 00015388 66BBC600            <1>       mov    bx, 0C6h ; 8 bit output (0C6h)
5121 0001538C 803D[A19B0100]02    <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
5122 00015393 7214                <1>       jb     short sb_play_2
5123 00015395 B720                <1>       mov    bh, 20h      ; 8 bit stereo (20h)
5124 00015397 EB10                <1>       jmp    short sb_play_2
5125                              <1> sb_play_1:
5126                              <1>       ; 16 bit samples
5127 00015399 66BBB610            <1>       mov    bx, 10B6h ; 16 bit output (0B6h)
5128 0001539D 803D[A19B0100]02    <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
5129 000153A4 7203                <1>       jb     short sb_play_2
5130 000153A6 80C720              <1>       add    bh, 20h      ; 16 bit stereo (30h)
5131                              <1> sb_play_2:
5132                              <1>       ; PCM output (8/16 bit mono autoinitialized transfer)
5133                              <1>       SbOut   bl ; bCommand
4945                              <2> %%Wait:
4946 000153A9 EC                  <2>  in al, dx
4947 000153AA 08C0                <2>  or al, al
4948 000153AC 78FB                <2>  js short %%Wait
4949 000153AE 88D8                <2>  mov al, %1
4950 000153B0 EE                  <2>  out dx, al
5134                              <1>       SbOut  bh ; bMode
4945                              <2> %%Wait:
4946 000153B1 EC                  <2>  in al, dx
4947 000153B2 08C0                <2>  or al, al
4948 000153B4 78FB                <2>  js short %%Wait
4949 000153B6 88F8                <2>  mov al, %1
4950 000153B8 EE                  <2>  out dx, al
5135 000153B9 8B1D[949B0100]      <1>       mov    ebx, [audio_dmabuff_size]  ; 15/05/2017
5136 000153BF D1EB                <1>       shr    ebx, 1 ; half buffer size
5137                              <1>       ; 20/10/2017
5138 000153C1 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit DMA
5139 000153C8 7502                <1>       jne    short sb_play_3
5140 000153CA D1EB                <1>       shr    ebx, 1 ; byte count to word count
5141                              <1> sb_play_3:
5142 000153CC 664B                <1>       dec    bx  ; wBlkSize is one less than the actual size
5143                              <1>       SbOut   bl
4945                              <2> %%Wait:
4946 000153CE EC                  <2>  in al, dx
4947 000153CF 08C0                <2>  or al, al
4948 000153D1 78FB                <2>  js short %%Wait
4949 000153D3 88D8                <2>  mov al, %1
4950 000153D5 EE                  <2>  out dx, al
```

```
5144                                <1>       SbOut   bh
4945                                <2> %%Wait:
4946 000153D6 EC                    <2>  in al, dx
4947 000153D7 08C0                  <2>  or al, al
4948 000153D9 78FB                  <2>  js short %%Wait
4949 000153DB 88F8                  <2>  mov al, %1
4950 000153DD EE                    <2>  out dx, al
5145                                <1>
5146 000153DE C605[A49B0100]01      <1>       mov   byte [audio_play_cmd], 1 ; playing !
5147                                <1>
5148                                <1>       ;; Set Voice and master volumes
5149                                <1>       ;mov   dx, [audio_io_base]
5150                                <1>       ;add   dl, 4 ; Mixer chip Register Address Port
5151                                <1>       ;SbOut 30h   ; select Master Volume Register (L)
5152                                <1>       ;inc   dl    ; Mixer chip Register Data Port
5153                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
5154                                <1>       ;dec   dl
5155                                <1>       ;SbOut 31h   ; select Master Volume Register (R)
5156                                <1>       ;inc   dl
5157                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
5158                                <1>       ;dec   dl
5159                                <1>       ;SbOut 32h   ; select Voice Volume Register (L)
5160                                <1>       ;inc   dl
5161                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
5162                                <1>       ;dec   dl
5163                                <1>       ;SbOut 33h   ; select Voice Volume Register (R)
5164                                <1>       ;inc   dl
5165                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
5166                                <1>       ;;
5167                                <1>       ;dec   dl
5168                                <1>       ;SbOut 44h   ; select Treble Register (L)
5169                                <1>       ;inc   dl
5170                                <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
5171                                <1>       ;dec   dl
5172                                <1>       ;SbOut 45h   ; select Treble Register (R)
5173                                <1>       ;inc   dl
5174                                <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
5175                                <1>       ;dec   dl
5176                                <1>       ;SbOut 46h   ; select Bass Register (L)
5177                                <1>       ;inc   dl
5178                                <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
5179                                <1>       ;dec   dl
5180                                <1>       ;SbOut 47h   ; select Bass Register (R)
5181                                <1>       ;inc   dl
5182                                <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
5183                                <1>
5184                                <1> sb_Exit:
5185                                <1>       ;popad
5186 000153E5 C3                   <1>       retn
5187                                <1>
5188                                <1> sb16_int_handler:
5189                                <1>       ; Interrupt Handler for Sound Blaster 16 Audio Card
5190                                <1>       ; Note: called by 'dev_IRQ_service'
5191                                <1>       ; 20/10/2017
5192                                <1>       ; 12/10/2017
5193                                <1>       ; 10/10/2017
5194                                <1>       ; 12/05/2017, 09/10/2017
5195                                <1>       ; 24/04/2017 (TRDOS 386 kernel, 'audio.s')
5196                                <1>       ; 10/03/2017 - 'PLAYWAV.PRG' ('playwav.s')
5197                                <1>
5198                                <1>       ;push  eax ; * must be saved !
5199                                <1>       ;push  ebx ; * must be saved !
5200                                <1>       ;push  ecx
5201                                <1>       ;push  edx
5202                                <1>       ;push  esi
5203                                <1>       ;push  edi
5204                                <1>
5205 000153E6 668B15[769B0100]     <1>       mov     dx, [audio_io_base]
5206                                <1>       ; 20/10/2017
5207 000153ED 80C20F                <1>       add     dl, 0Fh ; 2xFh (DSP 16 bit intr ack)
5208 000153F0 803D[A09B0100]10      <1>       cmp    byte [audio_bps], 16
5209 000153F7 7402                  <1>       je     short sb_irq_16bit_ack
5210                                <1> sb_irq_8bit_ack:
5211 000153F9 FECA                  <1>       dec    dl  ; 2xEh (DSP 8 bit intr ack)
5212                                <1> sb_irq_16bit_ack:
5213 000153FB EC                    <1>       in     al, dx
5214                                <1>
5215                                <1>       ;cmp   byte [audio_busy], 0
5216                                <1>       ;ja    short sb_irq_h3
5217                                <1>
5218                                <1>       ;mov   byte [audio_busy], 1
5219                                <1>
5220 000153FC 803D[A49B0100]01      <1>       cmp    byte [audio_play_cmd], 1
5221 00015403 7307                  <1>       jnb    short sb_irq_h1
5222                                <1> sb_irq_h0:
5223 00015405 E8A9000000            <1>       call   sb16_stop
5224 0001540A EB2B                  <1>       jmp    short sb_irq_h3
5225                                <1> sb_irq_h1:
5226                                <1>       ;call  sb16_tuneloop
5227                                <1>       ; 09/10/2017
5228                                <1> sb16_tuneloop:
5229 0001540C 8B3D[909B0100]        <1>       mov    edi, [audio_dma_buff]
5230 00015412 8B0D[949B0100]        <1>       mov    ecx, [audio_dmabuff_size]
5231 00015418 D1E9                  <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
5232                                <1>
5233                                <1>       ; 22/05/2017
5234 0001541A F605[989B0100]01      <1>       test   byte [audio_flag], 1  ; Current flag value
5235 00015421 7402                  <1>       jz     short sb_tlp1 ; EOL (Half Buffer 1 must be filled)
5236                                <1>       ; FLAG (Half Buffer 2 must be filled)
5237 00015423 01CF                  <1>       add    edi, ecx
5238                                <1>       ; 15/05/2017
5239                                <1> sb_tlp1:
5240 00015425 8B35[889B0100]        <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
5241                                <1>       ;rep   movsb
5242 0001542B C1E902                <1>       shr    ecx, 2 ; half buff size / 4
```

```
5243 0001542E F3A5                <1>       rep    movsd
5244                              <1>       ;retn
5245                              <1>
5246                              <1>       ; 10/10/2017
5247                              <1>       ; switch flag value
5248 00015430 8035[989B0100]01    <1>       xor    byte [audio_flag], 1
5249                              <1>
5250                              <1>       ; 12/10/2017
5251                              <1>       ; [audio_flag] = 0 : Playing dma half buffer 2 (odd intr count)
5252                              <1>                          ; Next buffer (to update) is dma half buff 1
5253                              <1>       ;             = 1 : Playing dma half buffer 1 (even intr count)
5254                              <1>                          ; Next buffer (to update) is dma half buff 2
5255                              <1>
5256                              <1> sb_irq_h3:
5257                              <1>       ;mov   byte [audio_busy], 0
5258                              <1>
5259                              <1>       ;pop   edi
5260                              <1>       ;pop   esi
5261                              <1>       ;pop   edx
5262                              <1>       ;pop   ecx
5263                              <1>       ;pop   ebx ; * must be restored !
5264                              <1>       ;pop   eax ; * must be restored !
5265                              <1>
5266 00015437 C3                  <1>       retn
5267                              <1>
5268                              <1> sb16_volume:
5269                              <1>       ; 22/10/2017
5270                              <1>       ; mov [audio_master_volume_l], cl
5271                              <1>       ; mov [audio_master_volume_h], ch
5272 00015438 66890D[A69B0100]    <1>       mov    [audio_master_volume], cx
5273                              <1> sb16_volume_initial:
5274 0001543F 6652                <1>       push   dx ; DX (port address) must be saved
5275 00015441 668B15[769B0100]    <1>       mov    dx, [audio_io_base]
5276 00015448 6683C204            <1>       add    dx, 4 ; Mixer chip address port
5277 0001544C B022                <1>       mov    al, 22h ; master volume
5278 0001544E EE                  <1>       out    dx, al
5279 0001544F 6642                <1>       inc    dx
5280 00015451 8A25[A69B0100]      <1>       mov    ah, [audio_master_volume_l]
5281 00015457 C0EC02              <1>       shr    ah, 2 ; 32 -> 8 level
5282 0001545A C0E405              <1>       shl    ah, 5 ; bit 5 to 7
5283 0001545D A0[A79B0100]        <1>       mov    al, [audio_master_volume_r]
5284 00015462 C0E802              <1>       shr    al, 2 ; 32 -> 8 level
5285                              <1>       ;and   al, 0Fh
5286 00015465 D0E0                <1>       shl    al, 1 ; bit 1 to 3
5287 00015467 08E0                <1>       or     al, ah
5288 00015469 EE                  <1>       out    dx, al
5289 0001546A 665A                <1>       pop    dx ; DX (port address) must be restored
5290 0001546C C3                  <1>       retn
5291                              <1>
5292                              <1> sb16_pause:
5293 0001546D 668B15[769B0100]    <1>       mov    dx, [audio_io_base]
5294 00015474 6683C20C            <1>       add    dx, 0Ch ; Command & Data Port
5295 00015478 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
5296 0001547F 7404                <1>       je     short sb_pause_1
5297                              <1>       ; 8 bit samples
5298 00015481 B3D0                <1>       mov    bl, 0D0h ; 8 bit DMA mode
5299 00015483 EB02                <1>       jmp    short sb_pause_2
5300                              <1> sb_pause_1:
5301                              <1>       ; 16 bit samples
5302 00015485 B3D5                <1>       mov    bl, 0D5h ; 16 bit DMA mode
5303                              <1> sb_pause_2:
5304                              <1>       SbOut   bl ; bCommand
4945                              <2> %%Wait:
4946 00015487 EC                  <2>  in al, dx
4947 00015488 08C0                <2>  or al, al
4948 0001548A 78FB                <2>  js short %%Wait
4949 0001548C 88D8                <2>  mov al, %1
4950 0001548E EE                  <2>  out dx, al
5305                              <1> sb_pause_3:
5306 0001548F C3                  <1>       retn
5307                              <1>
5308                              <1> sb16_continue:
5309 00015490 668B15[769B0100]    <1>       mov    dx, [audio_io_base]
5310 00015497 6683C20C            <1>       add    dx, 0Ch ; Command & Data Port
5311 0001549B 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
5312 000154A2 7404                <1>       je     short sb_cont_1
5313                              <1>       ; 8 bit samples
5314 000154A4 B3D4                <1>       mov    bl, 0D4h ; 8 bit DMA mode
5315 000154A6 EB02                <1>       jmp    short sb_cont_2
5316                              <1> sb_cont_1:
5317                              <1>       ; 16 bit samples
5318 000154A8 B3D6                <1>       mov    bl, 0D6h ; 16 bit DMA mode
5319                              <1> sb_cont_2:
5320                              <1>       SbOut   bl ; bCommand
4945                              <2> %%Wait:
4946 000154AA EC                  <2>  in al, dx
4947 000154AB 08C0                <2>  or al, al
4948 000154AD 78FB                <2>  js short %%Wait
4949 000154AF 88D8                <2>  mov al, %1
4950 000154B1 EE                  <2>  out dx, al
5321                              <1> sb_cont_3:
5322 000154B2 C3                  <1>       retn
5323                              <1>
5324                              <1> sb16_stop:
5325                              <1>       ; 24/04/2017
5326 000154B3 803D[A49B0100]00    <1>       cmp    byte [audio_play_cmd], 0
5327 000154BA 7648                <1>       jna    short sb16_stop_4
5328                              <1>
5329                              <1>       ; 22/05/2017
5330 000154BC 668B15[769B0100]    <1>       mov    dx, [audio_io_base]
5331 000154C3 6683C20C            <1>       add    dx, 0Ch
5332                              <1>
5333 000154C7 B3D9                <1>       mov    bl, 0D9h ; exit auto-initialize 16 bit transfer
5334                              <1>       ; stop  autoinitialized DMA transfer mode
5335 000154C9 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
```

```
5336 000154D0 7402                <1>       je     short sb16_stop_1
5337                              <1>       ;mov   bl, 0DAh ; exit auto-initialize 8 bit transfer
5338 000154D2 FEC3                <1>       inc    bl
5339                              <1> sb16_stop_1:
5340                              <1>       SbOut  bl ; exit auto-initialize transfer command
4945                              <2> %%Wait:
4946 000154D4 EC                  <2>  in al, dx
4947 000154D5 08C0                <2>  or al, al
4948 000154D7 78FB                <2>  js short %%Wait
4949 000154D9 88D8                <2>  mov al, %1
4950 000154DB EE                  <2>  out dx, al
5341                              <1>
5342 000154DC 30C0                <1>       xor     al, al ; stops all DMA processes on selected channel
5343                              <1>
5344 000154DE 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
5345 000154E5 7404                <1>       je     short sb16_stop_2
5346 000154E7 E60C                <1>       out    0Ch, al ; clear selected channel register
5347 000154E9 EB02                <1>       jmp    short sb16_stop_3
5348                              <1>
5349                              <1> sb16_stop_2:
5350 000154EB E6D8                <1>       out    0D8h, al ; clear selected channel register
5351                              <1>
5352                              <1> sb16_stop_3:
5353 000154ED C605[A49B0100]00    <1>       mov    byte [audio_play_cmd], 0 ; stop !
5354                              <1> SbDone:
5355                              <1>       ;mov   dx, [audio_io_base]
5356                              <1>       ;add   dx, 0Ch
5357                              <1>       SbOut   0D0h
4945                              <2> %%Wait:
4946 000154F4 EC                  <2>  in al, dx
4947 000154F5 08C0                <2>  or al, al
4948 000154F7 78FB                <2>  js short %%Wait
4949 000154F9 B0D0                <2>  mov al, %1
4950 000154FB EE                  <2>  out dx, al
5358                              <1>       SbOut   0D3h
4945                              <2> %%Wait:
4946 000154FC EC                  <2>  in al, dx
4947 000154FD 08C0                <2>  or al, al
4948 000154FF 78FB                <2>  js short %%Wait
4949 00015501 B0D3                <2>  mov al, %1
4950 00015503 EE                  <2>  out dx, al
5359                              <1> sb16_stop_4:
5360 00015504 C3                  <1>       retn
5361                              <1>
5362                              <1> sb16_reset:
5363                              <1>       ; 24/04/2017
5364 00015505 668B15[769B0100]    <1>       mov     dx, [audio_io_base] ; try to reset the DSP.
5365 0001550C 6683C206            <1>       add     dx, 06h
5366 00015510 B001                <1>       mov    al, 1
5367 00015512 EE                  <1>       out    dx, al
5368                              <1>
5369 00015513 EC                  <1>       in     al, dx
5370 00015514 EC                  <1>       in     al, dx
5371 00015515 EC                  <1>       in     al, dx
5372 00015516 EC                  <1>       in     al, dx
5373                              <1>
5374 00015517 30C0                <1>       xor     al, al
5375 00015519 EE                  <1>       out    dx, al
5376                              <1>
5377 0001551A 6683C208            <1>       add     dx, 08h
5378 0001551E 66B96400            <1>       mov    cx, 100
5379                              <1> sbrstWaitID:
5380 00015522 EC                  <1>       in     al, dx
5381 00015523 08C0                <1>       or      al, al
5382 00015525 7804                <1>       js      short sbrstGetID
5383 00015527 E2F9                <1>       loop    sbrstWaitID
5384 00015529 F9                  <1>       stc
5385 0001552A C3                  <1>       retn
5386                              <1> sbrstGetID:
5387 0001552B 6683EA04            <1>       sub     dx, 04h
5388 0001552F EC                  <1>       in     al, dx
5389 00015530 3CAA                <1>       cmp     al, 0AAh
5390 00015532 7406                <1>       je      short sb_rst_retn
5391 00015534 6683C204            <1>       add     dx, 04h
5392 00015538 E2E8                <1>       loop    sbrstWaitID
5393                              <1> sb_rst_retn:
5394 0001553A C3                  <1>       retn
5395                              <1>
5396                              <1> ac97_codec_config:
5397                              <1>       ; 10/06/2017
5398                              <1>       ; 05/06/2017
5399                              <1>       ; 29/05/2017
5400                              <1>       ; 28/05/2017 (TRDOS 386, 'audio.s')
5401                              <1>       ; 07/11/2016 (Erdogan Tan)
5402                              <1>       ; Derived from 'codecConfig' procedure in 'CODEC.ASM'
5403                              <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
5404                              <1>
5405                              <1>       ;; 'PLAYER.ASM'
5406                              <1>       ;; get ICH base address regs for mixer and bus master
5407                              <1>
5408                              <1> init_ac97_controller: ; 10/06/2017
5409 0001553B A1[789B0100]        <1>       mov    eax, [audio_dev_id]
5410                              <1>         ;mov al, NAMBAR_REG
5411                              <1>         ;;call  pciRegRead16                ; read PCI registers 10-11
5412                              <1>         ;call      pciRegRead32
5413                              <1>       ;and   dx, IO_ADDR_MASK          ; mask off BIT0
5414                              <1>       ;;and  edx, IO_ADDR_MASK
5415                              <1>
5416                              <1>         ;mov [NAMBAR], dx              ; save audio mixer base addr
5417                              <1>
5418                              <1>         ;mov    al, NABMBAR_REG
5419                              <1>         ;;call     pciRegRead16
5420                              <1>         ;call      pciRegRead32
5421                              <1>       ;and   dx, 0FFC0h ; IO_ADDR_MASK
5422                              <1>       ;;and  edx, 0FFC0h
```

```
                                <1>
                                <1>         ;mov    [NABMBAR], dx               ; save bus master base addr
                                <1>
                                <1>       ;mov   eax, [audio_dev_id]
00015540 B004                   <1>         mov     al, PCI_CMD_REG
                                <1>         ;call      pciRegRead8              ; read PCI command register
00015542 E84AF8FFFF             <1>         call pciRegRead16
00015547 80CA05                 <1>       or      dl, IO_ENA+BM_ENA              ; enable IO and bus master
                                <1>         ;call      pciRegWrite8
0001554A 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>
0001554F 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>       ;
00015554 66BA2C00               <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00015558 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
0001555F ED                     <1>       in     eax, dx
                                <1>       ; ?
00015560 66BA3000               <1>       mov    dx, GLOB_STS_REG ; 30h
00015564 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
0001556B ED                     <1>       in     eax, dx
                                <1>
0001556C 83F8FF                 <1>       cmp    eax, 0FFFFFFFFh ; -1
0001556F 744B                   <1>       je     short init_ac97_codec_err1
                                <1>
00015571 A900030010             <1>       test   eax, CTRL_ST_CREADY
00015576 7507                   <1>       jnz    short _ac97_codec_ready
                                <1>
00015578 E8EF020000             <1>       call   reset_ac97_codec
0001557D 723E                   <1>       jc     short init_ac97_codec_err2
                                <1>
                                <1> _ac97_codec_ready:
0001557F 668B15[A89B0100]       <1>       mov    dx, [NAMBAR]
                                <1>       ;add   dx, 0 ; ac_reg_0 ; reset register
00015586 66EF                   <1>       out    dx, ax
                                <1>
00015588 31C0                   <1>       xor    eax, eax ; 0
0001558A 668B15[A89B0100]       <1>       mov    dx, [NAMBAR]
00015591 6683C226               <1>       add    dx, CODEC_REG_POWERDOWN
00015595 66EF                   <1>       out    dx, ax
                                <1>
                                <1>       ; 10/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; wait for 1 second
00015597 B9E8030000             <1>       mov    ecx, 1000 ; 1000*0.25ms = 1s
                                <1> _ac97_codec_rloop:
0001559C E807F9FFFF             <1>       call   delay1_4ms
000155A1 E802F9FFFF             <1>       call   delay1_4ms
000155A6 E8FDF8FFFF             <1>       call   delay1_4ms
000155AB E8F8F8FFFF             <1>       call   delay1_4ms
                                <1>       ;mov   dx, [NAMBAR]
                                <1>       ;add   dx, CODEC_REG_POWERDOWN
000155B0 66ED                   <1>       in     ax, dx
000155B2 6683E00F               <1>       and    ax, 0Fh
000155B6 3C0F                   <1>       cmp    al, 0Fh
000155B8 7404                   <1>       je     short _ac97_codec_init_ok
000155BA E2E0                   <1>       loop   _ac97_codec_rloop
                                <1>
                                <1> init_ac97_codec_err1:
000155BC F9                     <1>       stc
                                <1> init_ac97_codec_err2:
000155BD C3                     <1>       retn
                                <1>
                                <1> _ac97_codec_init_ok:
```

```
5528 000155BE B002                <1>       mov    al, 2 ; force set 16-bit 2-channel PCM
5529 000155C0 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
5530 000155C4 660315[AA9B0100]    <1>       add    dx, [NABMBAR]
5531 000155CB EF                  <1>       out    dx, eax
5532                              <1>
5533                              <1>       ;call  delay1_4ms
5534                              <1>
5535                              <1>       ; 10/06/2017
5536 000155CC E849020000          <1>       call   reset_ac97_controller
5537                              <1>
5538                              <1> ;     call   setup_ac97_codec
5539                              <1> ;
5540                              <1> ;detect_ac97_codec:
5541                              <1> ;     retn
5542                              <1>
5543                              <1> setup_ac97_codec:
5544                              <1>       ; 22/07/2020
5545                              <1>       ; 10/06/2017
5546                              <1>       ; 29/05/2017
5547 000155D1 B802020000          <1>       mov    eax, 0202h
5548 000155D6 66A3[A69B0100]      <1>       mov    [audio_master_volume], ax
5549 000155DC 66B81F1F            <1>       mov    ax, 1F1Fh ; 31, 31
5550                              <1>
5551 000155E0 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5552 000155E7 6683C202            <1>       add    dx, CODEC_MASTER_VOL_REG  ;02h
5553 000155EB 6631C0              <1>       xor    ax, ax       ; volume attenuation = 0 (max. volume)
5554 000155EE 66EF                <1>       out    dx, ax
5555                              <1>
5556 000155F0 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5557 000155F7 6683C206            <1>       add    dx, CODEC_MASTER_MONO_VOL_REG    ;06h
5558                              <1>       ;xor    ax, ax
5559 000155FB 66EF                <1>       out    dx, ax
5560                              <1>
5561 000155FD 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5562 00015604 6683C20A            <1>       add    dx, CODEC_PCBEEP_VOL_REG  ;0Ah
5563                              <1>       ;xor   ax, ax
5564 00015608 66EF                <1>       out    dx, ax
5565                              <1>
5566 0001560A 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5567 00015611 6683C218            <1>       add    dx, CODEC_PCM_OUT_REG           ;18h
5568                              <1>       ;xor   ax, ax
5569 00015615 66EF                <1>       out    dx, ax
5570                              <1>
5571 00015617 66B80880            <1>       mov    ax, 8008h ; Mute
5572 0001561B 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5573                              <1>       ; 22/07/2020
5574 00015622 6683C20C            <1>       add    dx, CODEC_PHONE_VOL_REG          ;0Ch
5575                              <1>                           ; AC97_PHONE_VOL ; TAD Input (Mono)
5576 00015626 66EF                <1>       out    dx, ax
5577                              <1>
5578 00015628 66B80808            <1>       mov    ax, 0808h
5579 0001562C 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5580 00015633 6683C210            <1>       add    dx, CODEC_LINE_IN_VOL_REG ;10h ; Line Input (Stereo)
5581 00015637 66EF                <1>       out    dx, ax
5582                              <1>
5583                              <1>       ;mov   ax, 0808h
5584 00015639 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5585 00015640 6683C212            <1>       add    dx, CODEC_CD_VOL_REG ;12h ; CR Input (Stereo)
5586 00015644 66EF                <1>       out    dx, ax
5587                              <1>
5588                              <1>       ;mov   ax, 0808h
5589 00015646 668B15[A89B0100]    <1>       mov    dx, [NAMBAR]
5590 0001564D 6683C216            <1>       add    dx, CODEC_AUX_VOL_REG ;16h ; Aux Input (Stereo)
5591 00015651 66EF                <1>       out    dx, ax
5592                              <1>
5593                              <1>       ;call    delay1_4ms
5594                              <1>       ;call    delay1_4ms
5595                              <1>       ;call    delay1_4ms
5596                              <1>       ;call    delay1_4ms
5597                              <1>
5598                              <1> detect_ac97_codec:
5599 00015653 C3                  <1>       retn
5600                              <1>
5601                              <1> set_ac97_bdl: ; Set AC97 (ICH) Buffer Descriptor List
5602                              <1>       ; 17/06/2017
5603                              <1>       ; 11/06/2017
5604                              <1>       ; 28/05/2017
5605                              <1>       ; eax = dma buffer address = [audio_DMA_buff]
5606                              <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
5607                              <1>
5608 00015654 D1E9                <1>       shr    ecx, 1 ; dma half buffer size
5609 00015656 89CE                <1>       mov    esi, ecx
5610                              <1>
5611 00015658 BF[AC9B0100]        <1>       mov   edi, audio_bdl_buff ; get BDL address
5612 0001565D B910000000          <1>       mov   ecx, 32 / 2         ; make 32 entries in BDL
5613                              <1>
5614 00015662 EB05                <1>       jmp    short s_ac97_bdl1
5615                              <1>
5616                              <1> s_ac97_bdl0:
5617                              <1>       ; set buffer descriptor 0 to start of data file in memory
5618                              <1>
5619 00015664 A1[909B0100]        <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
5620                              <1>
5621                              <1> s_ac97_bdl1:
5622 00015669 AB                  <1>       stosd                    ; store dmabuffer1 address
5623                              <1>
5624 0001566A 89C2                <1>       mov    edx, eax
5625                              <1>
5626                              <1> ;
5627                              <1> ; Buffer Descriptors List
5628                              <1> ; As stated earlier, each buffer descriptor list is a set of (up to) 32
5629                              <1> ; descriptors, each 8 bytes in length. Bytes 0-3 of a descriptor entry point
5630                              <1> ; to a chunk of memory to either play from or record to. Bytes 4-7 of an
5631                              <1> ; entry describe various control things detailed below.
5632                              <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>
0001566C 89F0           <1>         mov     eax, esi ; DMA half buffer size
0001566E 01C2           <1>         add     edx, eax
00015670 D1E8           <1>         shr     eax, 1 ; count of 16 bit samples
                        <1>         ;or     eax, IOC+BUP
00015672 0D00000080     <1>         or      eax, IOC ; 11/06/2017
00015677 AB             <1>         stosd
                        <1>
                        <1> ; 2nd buffer:
                        <1>
00015678 89D0           <1>         mov     eax, edx ; Physical address of the 2nd half of DMA buffer
0001567A 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> ;
0001567B 89F0           <1>         mov     eax, esi ; DMA half buffer size
0001567D D1E8           <1>         shr     eax, 1 ; count of 16 bit samples
                        <1>         ;or     eax, IOC+BUP
0001567F 0D00000080     <1>         or      eax, IOC ; 11/06/2017
00015684 AB             <1>         stosd
                        <1>
00015685 E2DD           <1>         loop    s_ac97_bdl0
                        <1>
00015687 C3             <1>         retn
                        <1>
                        <1> ac97_start_play:
                        <1>         ; 28/05/2017
                        <1>         ; Derived from 'playWav' procedure in 'ICHWAV.ASM'
                        <1>         ; .wav player for DOS by Jeff Leyda (02/09/2002)
                        <1>
                        <1>         ; set output rate
                        <1>         ; entry: [audio_freq] = desired sample rate
                        <1>
00015688 668B15[A89B0100]   <1>         mov     dx, [NAMBAR]
0001568F 6683C22A           <1>         add     dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
00015693 66ED               <1>         in      ax, dx
00015695 6683C801           <1>         or      ax, 1
00015699 66EF               <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>
0001569B 66A1[A29B0100]     <1>         mov     ax, [audio_freq]          ; sample rate
                        <1>
000156A1 668B15[A89B0100]   <1>         mov     dx, [NAMBAR]
000156A8 6683C22C           <1>         add     dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
000156AC 66EF               <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> ;
                        <1> ; register reset the DMA engine. This may cause a pop noise on the output
                        <1> ; lines when the device is reset. Prolly a better idea to mute output, then
                        <1> ; reset.
                        <1> ;
000156AE 668B15[AA9B0100]   <1>         mov     dx, [NABMBAR]
000156B5 6683C21B           <1>         add     dx, PO_CR_REG                    ; set pointer to Cntl reg
000156B9 B002               <1>         mov     al, RR                           ; set reset
000156BB EE                 <1>         out     dx, al                           ; self clearing bit
                        <1> ;
                        <1> ;       mov     edi, audio_bdl_buff
                        <1> ;       mov     edx, [audio_dmabuff_size]
                        <1> ;       shr     edx, 1
                        <1> ;       mov     ecx, 32/2
                        <1> ;ac97_set_bdl_buffer:
                        <1> ;       ; 1st half of DMA buffer
                        <1> ;       mov     eax, [audio_dma_buff]
                        <1> ;       push    eax
                        <1> ;       stosd
                        <1> ;       mov     eax, edx ; dma buffer size / 2
                        <1> ;       or      eax, IOC+BUP
                        <1> ;       stosd
                        <1> ;       pop     eax
```

```
                        <1> ;       ; 2nd half of DMA buffer
                        <1> ;       add     eax, edx
                        <1> ;       stosd
                        <1> ;       mov     eax, edx ; dma buffer size / 2
                        <1> ;       or      eax, IOC+BUP
                        <1> ;       stosd
                        <1> ;       loop    ac97_set_bdl_buffer
                        <1>
                        <1> ; tell the DMA engine where to find our list of Buffer Descriptors.
                        <1> ; this 32bit value is a flat mode memory offset (ie no segment:offset)
                        <1> ;
                        <1> ; write NABMBAR+10h with offset of buffer descriptor list
                        <1> ;
000156BC B8[AC9B0100]   <1>         mov     eax, audio_bdl_buff
000156C1 668B15[AA9B0100]   <1>     mov     dx, [NABMBAR]
000156C8 6683C210       <1>         add     dx, PO_BDBAR_REG
000156CC EF             <1>         out     dx, eax
                        <1> ;
                        <1> ; All set. Let's play some music.
                        <1> ;
                        <1> ;
000156CD B81F000000     <1>         mov     eax, 31
000156D2 E816000000     <1>         call    set_ac97_LastValidIndex
                        <1>
000156D7 C605[A49B0100]01   <1>     mov     byte [audio_play_cmd], 1 ; play command (do not stop) !
                        <1>
                        <1> ac97_play: ; continue to play (after pause)
                        <1>         ; 11/06/2017
                        <1>         ; 29/05/2017
                        <1>         ; 28/05/2017
000156DE 668B15[AA9B0100]   <1>     mov     dx, [NABMBAR]
000156E5 6683C21B       <1>         add     dx, PO_CR_REG       ; PCM out control register
000156E9 B011           <1>         mov     al, IOCE+RPBM ; 29/05/2017
                        <1>         ;mov   al, 1Dh ; (Ref: KolibriOS, intelac97.asm, 'play:')
000156EB EE             <1>         out     dx, al              ; set start!
                        <1>
                        <1>         ;mov   byte [audio_play_cmd], 1 ; play command (do not stop) !
                        <1>
000156EC C3             <1>         retn
                        <1>
                        <1> ;input AL = index # to stop on
                        <1> set_ac97_LastValidIndex:
                        <1>         ; 28/05/2017
                        <1>         ; Derived from 'setLastValidIndex' procedure in 'ICHWAV.ASM'
                        <1>         ; .wav player for DOS by Jeff Leyda (02/09/2002)
000156ED 668B15[AA9B0100]   <1>     mov     dx, [NABMBAR]
000156F4 6683C215       <1>         add     dx, PO_LVI_REG
000156F8 EE             <1>         out     dx, al
                        <1>         ;mov   [audio_lvi], al ; for ac97_int_handler
000156F9 C3             <1>         retn
                        <1>
                        <1> ac97_volume:
                        <1>         ; 28/05/2017
                        <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>
000156FA 08DB           <1>         or      bl, bl
000156FC 7523           <1>         jnz     short ac97_vol_1 ; temporary !
000156FE 66B81F1F       <1>         mov     ax, 1F1Fh ; 31,31
00015702 38C1           <1>         cmp     cl, al
00015704 771B           <1>         ja      short ac97_vol_1 ; temporary !
00015706 38E5           <1>         cmp     ch, ah
00015708 7717           <1>         ja      short ac97_vol_1 ; temporary !
0001570A 66890D[A69B0100]   <1>     mov     [audio_master_volume], cx
00015711 6629C8         <1>         sub     ax, cx
00015714 668B15[A89B0100]   <1>     mov     dx, [NAMBAR]
0001571B 6683C202       <1>         add     dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
0001571F 66EF           <1>         out     dx, ax
                        <1> ac97_vol_1:
00015721 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>
00015722 66BA3000       <1>         mov     dx, GLOB_STS_REG
00015726 660315[AA9B0100]   <1>     add     dx, [NABMBAR]
0001572D ED             <1>         in      eax, dx
                        <1>
0001572E 83F8FF         <1>         cmp     eax, 0FFFFFFFFh ; -1
00015731 0F849A000000   <1>         je      _ac97_ih3 ; exit
                        <1>
00015737 A940000000     <1>         test    eax, 40h ; PCM Out Interrupt
0001573C 750E           <1>         jnz     short _ac97_ih0
                        <1>
0001573E 85C0           <1>         test    eax, eax
00015740 0F848B000000   <1>         jz      _ac97_ih3 ; exit
                        <1>
```

```
                                <1>       ;mov   dx, GLOB_STS_REG
                                <1>         ;add dx, [NABMBAR]
00015746 EF                     <1>       out    dx, eax
                                <1>
00015747 E985000000             <1>       jmp    _ac97_ih3 ; exit
                                <1>
                                <1> _ac97_ih0:
0001574C 50                     <1>       push   eax
                                <1>       ; 09/10/2017
0001574D 803D[A49B0100]01       <1>       cmp    byte [audio_play_cmd], 1
00015754 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>
00015756 66B81C00               <1>       mov    ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
0001575A 66BA1600               <1>       mov    dx, PO_SR_REG
0001575E 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015765 66EF                   <1>       out    dx, ax
                                <1>
00015767 66BA1400               <1>       mov    dx, PO_CIV_REG
0001576B 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015772 EC                     <1>       in     al, dx
                                <1>
                                <1>       ;cmp   al, [audio_civ] ; [audio_flag]
                                <1>       ;je    short _ac97_ih2
                                <1>
00015773 A2[A59B0100]           <1>       mov    [audio_civ], al
00015778 FEC8                   <1>       dec    al
                                <1>       ;inc   al ; 11/06/2017
0001577A 241F                   <1>       and    al, 1Fh
                                <1>
0001577C 66BA1500               <1>       mov    dx, PO_LVI_REG
00015780 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015787 EE                     <1>       out    dx, al
                                <1>
                                <1>       ; 12/10/2017
00015788 A0[A59B0100]           <1>       mov    al, [audio_civ]
0001578D FEC0                   <1>       inc    al
0001578F 2401                   <1>       and    al, 1
00015791 A2[989B0100]           <1>       mov    [audio_flag], al
                                <1>       ;; [audio_flag] : 0 = Buffer 1, 1 = Buffer 2
                                <1>       ;
00015796 58                     <1>       pop    eax
                                <1>       ;
00015797 83E040                 <1>       and    eax, 40h
0001579A 668B15[AA9B0100]       <1>       mov    dx, [NABMBAR]
000157A1 6683C230               <1>       add    dx, GLOB_STS_REG
000157A5 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
000157A6 8B3D[909B0100]         <1>       mov    edi, [audio_dma_buff]
000157AC 8B0D[949B0100]         <1>       mov    ecx, [audio_dmabuff_size]
000157B2 D1E9                   <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
                                <1>
                                <1>       ; 12/10/2017
000157B4 803D[989B0100]00       <1>       cmp    byte [audio_flag], 0
000157BB 7702                   <1>       ja     short _ac97_ih1  ; Playing Half Buffer 2 (Current: FLAG)
                                <1>       ; Playing Half Buffer 1 (Current: EOL)
000157BD 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>
000157BF 8B35[889B0100]         <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
000157C5 C1E902                 <1>       shr    ecx, 2 ; half buff size / 4
000157C8 F3A5                   <1>       rep    movsd
                                <1>
                                <1>       ; 10/10/2017
                                <1>       ; switch flag value
000157CA 8035[989B0100]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>
000157D1 C3                     <1>       retn
                                <1>
                                <1> _ac97_ih4:
                                <1>       ; 09/10/2017
000157D2 E818000000             <1>       call   _ac97_stop
                                <1>       ;
000157D7 58                     <1>       pop    eax
```

```
                                <1>       ;
000157D8 83E040                 <1>       and    eax, 40h
000157DB 668B15[AA9B0100]       <1>       mov    dx, [NABMBAR]
000157E2 6683C230               <1>       add    dx, GLOB_STS_REG
000157E6 EF                     <1>       out    dx, eax
                                <1>
                                <1>       ;; 13/06/2017
                                <1>       ;mov   al, 11h ; IOCE + RPBM
                                <1>       ;dx, PO_CR_REG
                                <1>       ;add   dx, [NABMBAR]
                                <1>       ;out   dx, al
                                <1>
                                <1>       ; 10/10/2017
                                <1>       ;jmp   short _ac97_ih3  ; exit
000157E7 C3                     <1>       retn
                                <1>
                                <1> ac97_stop:
                                <1>       ; 28/05/2017
000157E8 C605[A49B0100]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
000157EF 30C0                   <1>       xor    al, al ; 0
000157F1 E813000000             <1>       call   ac97_po_cmd
                                <1>
                                <1>       ; (Ref: KolibriOS, intelac97.asm, 'stop:')
                                <1>       ; Clear FIFOE, BCIS, LVBCI (Ref: Intel ICH hub manual)
000157F6 66B81C00               <1>       mov     ax, 1Ch
000157FA 668B15[AA9B0100]       <1>       mov     dx, [NABMBAR]
00015801 6683C216               <1>       add     dx, PO_SR_REG
00015805 66EF                   <1>       out     dx, ax
                                <1>
                                <1>       ;retn
                                <1>
                                <1>       ; 11/06/2017
00015807 B002                   <1>       mov     al, RR
                                <1> ac97_po_cmd:
                                <1>        ;11/06/2017
                                <1>       ; 29/05/2017
00015809 668B15[AA9B0100]       <1>       mov    dx, [NABMBAR]
00015810 6683C21B               <1>       add    dx, PO_CR_REG       ; PCM out control register
00015814 EE                     <1>       out    dx, al
00015815 C3                     <1>       retn
                                <1>
                                <1> ac97_pause:
                                <1>       ; 11/06/2017
                                <1>       ; 29/05/2017
00015816 B010                   <1>       mov    al, IOCE
00015818 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
0001581A 31C0                   <1>       xor    eax, eax
0001581C 66BA0B00               <1>       mov    dx, PI_CR_REG
00015820 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015827 EE                     <1>       out    dx, al
                                <1>
00015828 66BA1B00               <1>       mov    dx, PO_CR_REG
0001582C 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015833 EE                     <1>       out    dx, al
                                <1>
00015834 66BA2B00               <1>       mov    dx, MC_CR_REG
00015838 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
0001583F EE                     <1>       out    dx, al
                                <1>
00015840 B002                   <1>       mov    al, RR
00015842 66BA0B00               <1>       mov    dx, PI_CR_REG
00015846 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
0001584D EE                     <1>       out    dx, al
                                <1>
0001584E 66BA1B00               <1>       mov    dx, PO_CR_REG
00015852 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015859 EE                     <1>       out    dx, al
                                <1>
0001585A 66BA2B00               <1>       mov    dx, MC_CR_REG
0001585E 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015865 EE                     <1>       out    dx, al
                                <1>
00015866 C3                     <1>       retn
                                <1>
                                <1> ac97_reset:
                                <1>       ; 10/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 28/05/2017
00015867 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)
0001586C 66BA2C00               <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00015870 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015877 ED                     <1>       in     eax, dx
                                <1>
00015878 A902000000             <1>       test   eax, 2
0001587D 7407                   <1>       jz     short _r_ac97codec_cold
                                <1>
0001587F E80F000000             <1>       call   warm_ac97codec_reset
```

```
6053 00015884 7308              <1>       jnc    short _r_ac97codec_ok
6054                            <1> _r_ac97codec_cold:
6055 00015886 E83D000000        <1>       call   cold_ac97codec_reset
6056 0001588B 7301              <1>       jnc    short _r_ac97codec_ok
6057                            <1>
6058                            <1>       ; 16/04/2017
6059                            <1>         ;xor     eax, eax         ; timeout error
6060                            <1>             ;stc
6061 0001588D C3                <1>       retn
6062                            <1>
6063                            <1> _r_ac97codec_ok:
6064 0001588E 31C0              <1>         xor     eax, eax
6065                            <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
6066 00015890 FEC0              <1>         inc  al
6067 00015892 C3                <1>       retn
6068                            <1>
6069                            <1> warm_ac97codec_reset:
6070                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
6071 00015893 B806000000        <1>       mov   eax, 6
6072 00015898 66BA2C00          <1>       mov    dx, GLOB_CNT_REG ; 2Ch
6073 0001589C 660315[AA9B0100]  <1>       add    dx, [NABMBAR]
6074 000158A3 EF                <1>       out    dx, eax
6075                            <1>
6076 000158A4 B90A000000        <1>       mov    ecx, 10      ; total 1s
6077                            <1> _warm_ac97c_rst_wait:
6078 000158A9 51                <1>       push   ecx
6079 000158AA E8ECF5FFFF        <1>       call   delay_100ms
6080 000158AF 59                <1>       pop    ecx
6081                            <1>
6082 000158B0 66BA3000          <1>       mov    dx, GLOB_STS_REG ; 30h
6083 000158B4 660315[AA9B0100]  <1>       add    dx, [NABMBAR]
6084 000158BB ED                <1>       in     eax, dx
6085                            <1>
6086 000158BC A900030010        <1>       test   eax, CTRL_ST_CREADY
6087 000158C1 7504              <1>       jnz    short _warm_ac97c_rst_ok
6088                            <1>
6089 000158C3 49                <1>       dec   ecx
6090 000158C4 75E3              <1>       jnz   short _warm_ac97c_rst_wait
6091                            <1>
6092                            <1> _warm_ac97c_rst_fail:
6093 000158C6 F9                <1>         stc
6094                            <1> _warm_ac97c_rst_ok:
6095 000158C7 C3                <1>       retn
6096                            <1>
6097                            <1> cold_ac97codec_reset:
6098                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
6099 000158C8 B802000000        <1>       mov   eax, 2
6100 000158CD 66BA2C00          <1>       mov    dx, GLOB_CNT_REG ; 2Ch
6101 000158D1 660315[AA9B0100]  <1>       add    dx, [NABMBAR]
6102 000158D8 EF                <1>       out    dx, eax
6103                            <1>
6104 000158D9 E8BDF5FFFF        <1>       call   delay_100ms  ; wait 100 ms
6105 000158DE E8B8F5FFFF        <1>       call   delay_100ms  ; wait 100 ms
6106 000158E3 E8B3F5FFFF        <1>       call   delay_100ms  ; wait 100 ms
6107 000158E8 E8AEF5FFFF        <1>       call   delay_100ms  ; wait 100 ms
6108                            <1>
6109 000158ED B910000000        <1>       mov    ecx, 16      ; total 20*100 ms = 2s
6110                            <1> _cold_ac97c_rst_wait:
6111 000158F2 66BA3000          <1>       mov    dx, GLOB_STS_REG ; 30h
6112 000158F6 660315[AA9B0100]  <1>       add    dx, [NABMBAR]
6113 000158FD ED                <1>       in     eax, dx
6114                            <1>
6115 000158FE A900030010        <1>       test   eax, CTRL_ST_CREADY
6116 00015903 750B              <1>       jnz    short _cold_ac97c_rst_ok
6117                            <1>
6118 00015905 51                <1>       push   ecx
6119 00015906 E890F5FFFF        <1>       call   delay_100ms
6120 0001590B 59                <1>       pop    ecx
6121                            <1>
6122 0001590C 49                <1>       dec   ecx
6123 0001590D 75E3              <1>       jnz   short _cold_ac97c_rst_wait
6124                            <1>
6125                            <1> _cold_ac97c_rst_fail:
6126 0001590F F9                <1>       stc
6127                            <1> _cold_ac97c_rst_ok:
6128 00015910 C3                <1>       retn
6129                            <1>
6130                            <1> sb16_current_sound_data:
6131                            <1>       ; 20/08/2017
6132                            <1>       ; 24/06/2017
6133                            <1>       ; 22/06/2017
6134                            <1>       ; get current sound (PCM out) data for graphics
6135                            <1>       ; (for Sound Blaster 16)
6136                            <1>       ; ebx = Physical address (on page boundary)
6137                            <1>       ; ecx = Byte count
6138                            <1>       ; [audio_buff_size]
6139                            <1>
6140                            <1>       ;;mov  edi, [audio_buff_size]
6141                            <1>       ;mov   edi, [audio_dmabuff_size]
6142                            <1>       ;mov   esi, [audio_dma_buff]
6143 00015911 39CF              <1>       cmp    edi, ecx
6144 00015913 7302              <1>       jnb    short sb16_gcd_0
6145 00015915 89F9              <1>       mov    ecx, edi
6146                            <1> sb16_gcd_0:
6147                            <1>       ; 20/08/2017
6148 00015917 803D[A09B0100]10  <1>       cmp    byte [audio_bps], 16
6149 0001591E 750F              <1>       jne    short sb16_gcd_1 ; 8 bit DMA channel
6150 00015920 E4C6              <1>       in     al, 0C6h ; DMA channel 5 count register
6151 00015922 88C2              <1>       mov    dl, al
6152 00015924 E4C6              <1>       in     al, 0C6h
6153 00015926 88C6              <1>       mov    dh, al
6154 00015928 0FB7C2            <1>       movzx  eax, dx
6155 0001592B D1E0              <1>       shl    eax, 1 ; word count -> byte count
6156 0001592D EB4E              <1>       jmp    short sb16_gcd_2
6157                            <1> sb16_gcd_1:
```

```
6158 0001592F E403                <1>       in     al, 03h ; DMA channel 1 count register
6159 00015931 88C2                <1>       mov    dl, al
6160 00015933 E403                <1>       in     al, 03h
6161 00015935 88C6                <1>       mov    dh, al
6162 00015937 0FB7C2              <1>       movzx  eax, dx
6163 0001593A EB41                <1>       jmp    short sb16_gcd_2
6164                              <1> ;sb16_gcd_2:
6165                              <1> ;     cmp    eax, ecx
6166                              <1> ;     jnb    short sb16_gcd_3
6167                              <1> ;     ; remain count < graphics bytes
6168                              <1> ;     mov    eax, ecx ; fix remain count to data size
6169                              <1> ;sb16_gcd_3:
6170                              <1> ;     sub    edi, eax
6171                              <1> ;     jna    short sb16_gcd_4
6172                              <1> ;     add    esi, edi ; dma buffer offset
6173                              <1> ;sb16_gcd_4:
6174                              <1> ;     mov    edi, ebx ; buffer address (for graphics)
6175                              <1> ;     mov    [u.r0], ecx
6176                              <1> ;     rep    movsb
6177                              <1> ;     retn
6178                              <1>
6179                              <1> get_current_sound_data:
6180                              <1>       ; 24/06/2017
6181                              <1>       ; 22/06/2017
6182                              <1>       ; get current sound (PCM out) data for graphics
6183                              <1>       ;
6184                              <1>       ; ebx = Physical address (on page boundary)
6185                              <1>       ; ecx = Byte count
6186                              <1>       ; [audio_buff_size]
6187                              <1>
6188                              <1>       ;mov   edi, [audio_buff_size]
6189 0001593C 8B3D[949B0100]      <1>       mov    edi, [audio_dmabuff_size]
6190 00015942 8B35[909B0100]      <1>       mov    esi, [audio_dma_buff]
6191 00015948 803D[719B0100]02    <1>       cmp    byte [audio_device], 2
6192 0001594F 72C0                <1>       jb     short sb16_current_sound_data ; = 1
6193 00015951 D1EF                <1>       shr    edi, 1
6194 00015953 39CF                <1>       cmp    edi, ecx
6195 00015955 7302                <1>       jnb    short gcd_0
6196 00015957 89F9                <1>       mov    ecx, edi
6197                              <1> gcd_0:
6198 00015959 803D[719B0100]03    <1>       cmp    byte [audio_device], 3
6199 00015960 7232                <1>       jb     short ac97_current_sound_data ; = 2
6200                              <1>       ; = 3
6201                              <1> vt8233_current_sound_data:
6202                              <1>       ; 22/06/2017
6203                              <1>       ; 21/06/2017
6204                              <1>       ; get current sound (PCM out) data for graphics
6205                              <1>       ; (for VT 8233, VT 8237R)
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                              <1>       ;mov   edi, [audio_dmabuff_size]
6212                              <1>       ;mov   esi, [audio_dma_buff]
6213                              <1>       ;shr   edi, 1
6214                              <1>       ;cmp   edi, ecx
6215                              <1>       ;jnb   short vt8233_gcd_1
6216                              <1>       ;mov   ecx, edi
6217                              <1> vt8233_gcd_1:
6218 00015962 BA0C000000          <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
6219 00015967 E88FF5FFFF          <1>       call   ctrl_io_r32
6220 0001596C 89C2                <1>       mov    edx, eax ; remain count (bits 23-0),
6221                              <1>                       ; SGD index (bits 31-24)
6222 0001596E 81E200000001        <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
6223 00015974 7402                <1>       jz     short vt8233_gcd_2
6224                              <1>       ; the second half of DMA buffer
6225 00015976 01FE                <1>       add    esi, edi
6226                              <1> vt8233_gcd_2:
6227 00015978 25FFFFFF00          <1>       and    eax, 0FFFFFFh ; bits 23-0
6228                              <1> ac97_gcd_2:
6229                              <1> sb16_gcd_2:
6230 0001597D 39C8                <1>       cmp    eax, ecx
6231 0001597F 7302                <1>       jnb    short vt8233_gcd_3
6232                              <1>       ; remain count < graphics bytes
6233 00015981 89C8                <1>       mov    eax, ecx ; fix remain count to data size
6234                              <1> vt8233_gcd_3:
6235 00015983 29C7                <1>       sub    edi, eax
6236 00015985 7602                <1>       jna    short vt8233_gcd_4
6237 00015987 01FE                <1>       add    esi, edi ; dma buffer offset
6238                              <1> vt8233_gcd_4:
6239 00015989 89DF                <1>       mov    edi, ebx ; buffer address (for graphics)
6240 0001598B 890D[64030300]      <1>       mov    [u.r0], ecx
6241 00015991 F3A4                <1>       rep    movsb
6242                              <1> vt8233_gcd_5:
6243 00015993 C3                  <1>       retn
6244                              <1>
6245                              <1> ac97_current_sound_data:
6246                              <1>       ; 23/06/2017
6247                              <1>       ; 22/06/2017
6248                              <1>       ; get current sound (PCM out) data for graphics
6249                              <1>       ; (for AC'97, ICH)
6250                              <1>       ; ebx = Physical address (on page boundary)
6251                              <1>       ; ecx = Byte count
6252                              <1>       ; [audio_buff_size]
6253                              <1>
6254                              <1>       ;;mov  edi, [audio_buff_size]
6255                              <1>       ;mov   edi, [audio_dmabuff_size]
6256                              <1>       ;mov   esi, [audio_dma_buff]
6257                              <1>       ;shr   edi, 1
6258                              <1>       ;cmp   edi, ecx
6259                              <1>       ;jnb   short ac97_gcd_0
6260                              <1>       ;mov   ecx, edi
6261                              <1> ac97_gcd_0:
6262 00015994 66BA1400            <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
```

```
6263 00015998 660315[AA9B0100]    <1>       add    dx, [NABMBAR]
6264 0001599F EC                  <1>       in     al, dx ; current index value
6265 000159A0 A801                <1>       test   al, 1
6266 000159A2 7402                <1>       jz     short ac97_gcd_1
6267 000159A4 01FE                <1>       add    esi, edi
6268                              <1> ac97_gcd_1:
6269 000159A6 31C0                <1>       xor    eax, eax
6270 000159A8 66BA1800            <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
6271 000159AC 660315[AA9B0100]    <1>       add    dx, [NABMBAR]
6272 000159B3 66ED                <1>       in     ax, dx ; remain dwords
6273 000159B5 C1E002              <1>       shl    eax, 2 ; remain bytes ; 23/06/2017
6274 000159B8 EBC3                <1>       jmp    short ac97_gcd_2
6275                              <1> ;      cmp    eax, ecx
6276                              <1> ;      jnb    short ac97_gcd_2
6277                              <1> ;      ; remain count < graphics bytes
6278                              <1> ;      mov    eax, ecx ; fix remain count to data size
6279                              <1> ;ac97_gcd_2:
6280                              <1> ;      sub    edi, eax
6281                              <1> ;      jna    short ac97_gcd_3
6282                              <1> ;      add    esi, edi ; dma buffer offset
6283                              <1> ;ac97_gcd_3:
6284                              <1> ;      mov    edi, ebx ; buffer address (for graphics)
6285                              <1> ;      mov    [u.r0], ecx
6286                              <1> ;      rep    movsb
6287                              <1> ;      retn
6288                              <1>
6289                              <1> sb16_get_dma_buff_off:
6290                              <1>       ; 28/10/2017
6291                              <1>       ; 24/06/2017
6292                              <1>       ; 22/06/2017
6293                              <1>       ; get current (PCM OUT DMA buffer) pointer
6294                              <1>       ; (for Sound Blaster 16)
6295                              <1>
6296                              <1>       ;mov   ecx, [audio_dmabuff_size]
6297                              <1>       ;xor   ebx, ebx
6298                              <1>       ;shr   ecx, 1
6299                              <1> sb16_gdmabo_0:
6300                              <1>       ; 28/10/2017
6301 000159BA 803D[A09B0100]10    <1>       cmp    byte [audio_bps], 16
6302 000159C1 750F                <1>       jne    short sb16_gdmabo_1 ; 8 bit DMA channel
6303                              <1>       ; 16 bit DMA channel
6304 000159C3 E4C6                <1>       in     al, 0C6h ; DMA channel 5 count register
6305 000159C5 88C2                <1>       mov    dl, al
6306 000159C7 E4C6                <1>       in     al, 0C6h
6307 000159C9 88C6                <1>       mov   dh, al
6308 000159CB 0FB7C2              <1>       movzx  eax, dx
6309 000159CE D1E0                <1>       shl    eax, 1 ; word count -> byte count
6310 000159D0 EB3D                <1>       jmp    short sb16_gdmabo_2
6311                              <1> sb16_gdmabo_1:
6312 000159D2 E403                <1>       in     al, 03h ; DMA channel 1 count register
6313 000159D4 88C2                <1>       mov    dl, al
6314 000159D6 E403                <1>       in     al, 03h
6315 000159D8 88C6                <1>       mov   dh, al
6316 000159DA 0FB7C2              <1>       movzx  eax, dx
6317 000159DD EB30                <1>       jmp    short sb16_gdmabo_2
6318                              <1>
6319                              <1> get_dma_buffer_offset:
6320                              <1>       ; 24/06/2017
6321                              <1>       ; 22/06/2017
6322                              <1>       ; get current sound (PCM out) data for graphics
6323                              <1>       ;
6324                              <1>       ; ebx = Physical address (on page boundary)
6325                              <1>       ; ecx = Byte count
6326                              <1>       ; [audio_buff_size]
6327                              <1>
6328 000159DF 8B0D[949B0100]      <1>       mov    ecx, [audio_dmabuff_size]
6329 000159E5 31DB                <1>       xor    ebx, ebx
6330                              <1> gdmabo_0:
6331 000159E7 803D[719B0100]02    <1>       cmp    byte [audio_device], 2
6332 000159EE 72CA                <1>       jb     short sb16_get_dma_buff_off
6333 000159F0 742A                <1>       je     short ac97_get_dma_buff_off
6334                              <1>
6335                              <1> vt8233_get_dma_buff_off:
6336                              <1>       ; 24/06/2017
6337                              <1>       ; 22/06/2017
6338                              <1>       ; get current (PCM OUT DMA buffer) pointer
6339                              <1>       ; (for VT 8233, VT 8237R)
6340                              <1>
6341                              <1>       ;mov   ecx, [audio_dmabuff_size]
6342                              <1>       ;xor   ebx, ebx
6343 000159F2 D1E9                <1>       shr    ecx, 1
6344                              <1> vt8233_gdmabo_0:
6345 000159F4 BA0C000000          <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
6346 000159F9 E8FDF4FFFF          <1>       call   ctrl_io_r32
6347 000159FE 89C2                <1>       mov    edx, eax ; remain count (bits 23-0),
6348                              <1>                       ; SGD index (bits 31-24)
6349 00015A00 81E200000001        <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
6350 00015A06 7402                <1>       jz     short vt8233_gdmabo_1
6351                              <1>       ; the second half of DMA buffer
6352 00015A08 89CB                <1>       mov    ebx, ecx
6353                              <1> vt8233_gdmabo_1:
6354 00015A0A 25FFFFFF00          <1>       and    eax, 0FFFFFFh ; bits 23-0
6355                              <1> sb16_gdmabo_2:
6356                              <1> ac97_gdmabo_2:
6357 00015A0F 29C1                <1>       sub    ecx, eax
6358 00015A11 7602                <1>       jna    short vt8233_gdmabo_2
6359 00015A13 01CB                <1>       add    ebx, ecx ; dma buffer offset
6360                              <1> vt8233_gdmabo_2:
6361 00015A15 891D[64030300]      <1>       mov    [u.r0], ebx
6362 00015A1B C3                  <1>       retn
6363                              <1>
6364                              <1> ac97_get_dma_buff_off:
6365                              <1>       ; 24/06/2017
6366                              <1>       ; 22/06/2017
6367                              <1>       ; get current (PCM OUT DMA buffer) pointer
```

```
                                <1>       ; (for AC'97, ICH)
                                <1>       ; ebx = Physical address (on page boundary)
                                <1>       ; ecx = Byte count
                                <1>       ; [audio_buff_size]
                                <1>
                                <1>       ;mov   ecx, [audio_dmabuff_size]
                                <1>       ;xor   ebx, ebx
00015A1C D1E9                   <1>       shr    ecx, 1
                                <1> ac97_gdmabo_0:
00015A1E 66BA1400               <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
00015A22 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015A29 EC                     <1>       in     al, dx ; current index value
00015A2A A801                   <1>       test   al, 1
00015A2C 7402                   <1>       jz     short ac97_gdmabo_1
00015A2E 89CB                   <1>       mov    ebx, ecx
                                <1> ac97_gdmabo_1:
00015A30 31C0                   <1>       xor    eax, eax
00015A32 66BA1800               <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
00015A36 660315[AA9B0100]       <1>       add    dx, [NABMBAR]
00015A3D 66ED                   <1>       in     ax, dx ; remain dwords
00015A3F EBCE                   <1>       jmp    short ac97_gdmabo_2
                                
00015A41 90<rep 3h>                 align 4
                                
                                    %include 'vgadata.s' ; 04/07/2016
                               <1> ; ****************************************************************************************************
                               <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - vgadata.s  (palette and font data)
                               <1> ; ----------------------------------------------------------------------------
                               <1> ; Last Update: 01/01/2021
                               <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 'Plex86/Bochs VGABios' source code, vgabios-0.7a (2011)
                               <1> ; by the LGPL VGABios Developers Team (2001-2008), 'vgatables.h'
                               <1> ;
                               <1> ; Oracle VirtualBox 5.0.24 VGABios Source Code
                               <1> ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                               <1> ;
                               <1> ; Palette and font data in assembly language format:
                               <1> ; 'VBoxVgaBiosAlternative.asm'
                               <1> 
                               <1> ; ****************************************************************************************************
                               <1> 
                               <1> ; 25/11/2020 (TRDOS 386 v2.0.3)
                               <1> ; ('vgatables.h' - 30/12/2019 - vruppert)
                               <1> 
                               <1> ; 04/07/2016
                               <1> ; COLOR DATA
                               <1> 
                               <1> palette0: ; (63+1)*3
00015A44 000000000000000000-   <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00015A4D 00000000000000        <1>
00015A54 00000000000000002A-   <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
00015A5D 2A2A2A2A2A2A          <1>
00015A64 2A2A2A2A2A2A2A2A2A-   <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
00015A6D 2A2A2A2A2A2A          <1>
00015A74 2A2A2A2A2A2A2A2A2A-   <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
00015A7D 2A2A2A2A2A2A          <1>
00015A84 2A2A2A2A2A2A2A2A3F-   <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
00015A8D 3F3F3F3F3F3F3F        <1>
00015A94 3F3F3F3F3F3F3F3F3F-   <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
00015A9D 3F3F3F3F3F3F3F        <1>
00015AA4 000000000000000000-   <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00015AAD 00000000000000        <1>
00015AB4 00000000000000002A-   <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
00015ABD 2A2A2A2A2A2A          <1>
00015AC4 2A2A2A2A2A2A2A2A2A-   <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
00015ACD 2A2A2A2A2A2A          <1>
00015AD4 2A2A2A2A2A2A2A2A2A-   <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
00015ADD 2A2A2A2A2A2A          <1>
00015AE4 2A2A2A2A2A2A2A2A3F-   <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
00015AED 3F3F3F3F3F3F          <1>
00015AF4 3F3F3F3F3F3F3F3F3F-   <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
00015AFD 3F3F3F3F3F3F          <1>
                               <1> palette1: ; (63+1)*3
00015B04 00000000002A002A00-   <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
00015B0D 002A2A2A00002A        <1>
00015B14 002A2A15002A2A2A00-   <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
00015B1D 000000002A002A        <1>
00015B24 00002A2A2A00002A00-   <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
00015B2D 2A2A15002A2A2A        <1>
00015B34 15151515153F153F15-   <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
00015B3D 153F3F3F15153F        <1>
00015B44 153F3F3F153F3F3F15-   <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
00015B4D 151515153F153F        <1>
00015B54 15153F3F3F15153F15-   <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
00015B5D 3F3F3F153F3F3F        <1>
00015B64 00000000002A002A00-   <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
00015B6D 002A2A2A00002A        <1>
00015B74 002A2A15002A2A2A00-   <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
00015B7D 000000002A002A        <1>
00015B84 00002A2A2A00002A00-   <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
00015B8D 2A2A15002A2A2A        <1>
00015B94 15151515153F153F15-   <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
00015B9D 153F3F3F15153F        <1>
00015BA4 153F3F3F153F3F3F15-   <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
00015BAD 151515153F153F        <1>
00015BB4 15153F3F3F15153F15-   <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
00015BBD 3F3F3F153F3F3F        <1>
                               <1> palette2: ; (63+1)*3
00015BC4 00000000002A002A00-   <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
00015BCD 002A2A2A00002A        <1>
00015BD4 002A2A2A002A2A2A00-   <1>     db  000h, 02ah, 02ah, 02ah, 000h, 02ah, 02ah, 02ah, 000h, 000h, 015h, 000h, 000h, 03fh, 000h, 02ah
00015BDD 001500003F002A        <1>
```

```
3492 00015BE4 15002A3F2A00152A00- <1>     db  015h, 000h, 02ah, 03fh, 02ah, 000h, 015h, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 02ah, 02ah, 03fh
3492 00015BED 3F2A2A152A2A3F      <1>
3493 00015BF4 00150000152A003F00- <1>     db  000h, 015h, 000h, 000h, 015h, 02ah, 000h, 03fh, 000h, 000h, 03fh, 02ah, 02ah, 015h, 000h, 02ah
3493 00015BFD 003F2A2A15002A      <1>
3494 00015C04 152A2A3F002A3F2A00- <1>     db  015h, 02ah, 02ah, 03fh, 000h, 02ah, 03fh, 02ah, 000h, 015h, 015h, 000h, 015h, 03fh, 000h, 03fh
3494 00015C0D 151500153F003F      <1>
3495 00015C14 15003F3F2A15152A15- <1>     db  015h, 000h, 03fh, 03fh, 02ah, 015h, 015h, 02ah, 015h, 03fh, 02ah, 03fh, 015h, 02ah, 03fh, 03fh
3495 00015C1D 3F2A3F152A3F3F      <1>
3496 00015C24 15000015002A152A00- <1>     db  015h, 000h, 000h, 015h, 000h, 02ah, 015h, 02ah, 000h, 015h, 02ah, 02ah, 03fh, 000h, 000h, 03fh
3496 00015C2D 152A2A3F00003F      <1>
3497 00015C34 002A3F2A003F2A2A15- <1>     db  000h, 02ah, 03fh, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 000h, 015h, 015h, 000h, 03fh, 015h, 02ah
3497 00015C3D 001515003F152A      <1>
3498 00015C44 15152A3F3F00153F00- <1>     db  015h, 015h, 02ah, 03fh, 03fh, 000h, 015h, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 03fh, 02ah, 03fh
3498 00015C4D 3F3F2A153F2A3F      <1>
3499 00015C54 15150015152A153F00- <1>     db  015h, 015h, 000h, 015h, 015h, 02ah, 015h, 03fh, 000h, 015h, 03fh, 02ah, 03fh, 015h, 000h, 03fh
3499 00015C5D 153F2A3F15003F      <1>
3500 00015C64 152A3F3F003F3F2A15- <1>     db  015h, 02ah, 03fh, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
3500 00015C6D 151515153F153F      <1>
3501 00015C74 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
3501 00015C7D 3F3F3F153F3F3F      <1>
3502                              <1> palette3: ; 256*3
3503 00015C84 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
3503 00015C8D 002A2A2A00002A      <1>
3504 00015C94 002A2A15002A2A2A15- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
3504 00015C9D 151515153F153F      <1>
3505 00015CA4 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
3505 00015CAD 3F3F3F153F3F3F      <1>
3506 00015CB4 0000000505050808- <1>     db  000h, 000h, 000h, 005h, 005h, 005h, 008h, 008h, 008h, 00bh, 00bh, 00bh, 00eh, 00eh, 00eh, 011h
3506 00015CBD 0B0B0B0E0E0E11      <1>
3507 00015CC4 11111414141818181C- <1>     db  011h, 011h, 014h, 014h, 014h, 018h, 018h, 018h, 01ch, 01ch, 01ch, 020h, 020h, 020h, 024h, 024h
3507 00015CCD 1C1C2020202424      <1>
3508 00015CD4 242828282D2D2D3232- <1>     db  024h, 028h, 028h, 028h, 02dh, 02dh, 02dh, 032h, 032h, 032h, 038h, 038h, 038h, 03fh, 03fh, 03fh
3508 00015CDD 323838383F3F3F      <1>
3509 00015CE4 00003F10003F1F003F- <1>     db  000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 03fh, 03fh
3509 00015CED 2F003F3F003F3F      <1>
3510 00015CF4 002F3F001F3F00103F- <1>     db  000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh
3510 00015CFD 00003F10003F1F      <1>
3511 00015D04 003F2F003F3F002F3F- <1>     db  000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h
3511 00015D0D 001F3F00103F00      <1>
3512 00015D14 003F00003F10003F1F- <1>     db  000h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h
3512 00015D1D 003F2F003F3F00      <1>
3513 00015D24 2F3F001F3F00103F1F- <1>     db  02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh
3513 00015D2D 1F3F271F3F2F1F      <1>
3514 00015D34 3F371F3F3F1F3F3F1F- <1>     db  03fh, 037h, 01fh, 03fh, 03fh, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h
3514 00015D3D 373F1F2F3F1F27      <1>
3515 00015D44 3F1F1F3F271F3F2F1F- <1>     db  03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h
3515 00015D4D 3F371F3F3F1F37      <1>
3516 00015D54 3F1F2F3F1F273F1F1F- <1>     db  03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh, 01fh, 01fh, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh
3516 00015D5D 3F1F1F3F271F3F      <1>
3517 00015D64 2F1F3F371F3F3F1F37- <1>     db  02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh
3517 00015D6D 3F1F2F3F1F273F      <1>
3518 00015D74 2D2D3F312D3F362D3F- <1>     db  02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03fh, 03fh
3518 00015D7D 3A2D3F3F2D3F3F      <1>
3519 00015D84 2D3A3F2D363F2D313F- <1>     db  02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h
3519 00015D8D 2D2D3F312D3F36      <1>
3520 00015D94 2D3F3A2D3F3F2D3A3F- <1>     db  02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh
3520 00015D9D 2D363F2D313F2D      <1>
3521 00015DA4 2D3F2D2D3F312D3F36- <1>     db  02dh, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh
3521 00015DAD 2D3F3A2D3F3F2D      <1>
3522 00015DB4 3A3F2D363F2D313F00- <1>     db  03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h
3522 00015DBD 001C07001C0E00      <1>
3523 00015DC4 1C15001C1C001C1C00- <1>     db  01ch, 015h, 000h, 01ch, 01ch, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h
3523 00015DCD 151C000E1C0007      <1>
3524 00015DD4 1C00001C07001C0E00- <1>     db  01ch, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h
3524 00015DDD 1C15001C1C0015      <1>
3525 00015DE4 1C000E1C00071C0000- <1>     db  01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch, 000h, 000h, 01ch, 000h, 000h, 01ch, 007h, 000h, 01ch
3525 00015DED 1C00001C07001C      <1>
3526 00015DF4 0E001C15001C1C0015- <1>     db  00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch
3526 00015DFD 1C000E1C00071C      <1>
3527 00015E04 0E0E1C110E1C150E1C- <1>     db  00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 01ch, 01ch
3527 00015E0D 180E1C1C0E1C1C      <1>
3528 00015E14 0E181C0E151C0E111C- <1>     db  00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h
3528 00015E1D 0E0E1C110E1C15      <1>
3529 00015E24 0E1C180E1C1C0E181C- <1>     db  00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh
3529 00015E2D 0E151C0E111C0E      <1>
3530 00015E34 0E1C0E0E1C110E1C15- <1>     db  00eh, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh
3530 00015E3D 0E1C180E1C1C0E      <1>
3531 00015E44 181C0E151C0E111C14- <1>     db  018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h
3531 00015E4D 141C16141C1814      <1>
3532 00015E54 1C1A141C1C141C1C14- <1>     db  01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h
3532 00015E5D 1A1C14181C1416      <1>
3533 00015E64 1C14141C16141C1814- <1>     db  01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah
3533 00015E6D 1C1A141C1C141A      <1>
3534 00015E74 1C14181C14161C1414- <1>     db  01ch, 014h, 018h, 01ch, 014h, 016h, 01ch, 014h, 014h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch
3534 00015E7D 1C14141C16141C      <1>
3535 00015E84 18141C1A141C1C141A- <1>     db  018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h, 01ch
3535 00015E8D 1C14181C14161C      <1>
3536 00015E94 0000100400100800 10- <1>     db  000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h, 010h, 010h
3536 00015E9D 0C001010001010      <1>
3537 00015EA4 000C10000810000410- <1>     db  000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h
3537 00015EAD 00001004001008      <1>
3538 00015EB4 00100C001010000C10- <1>     db  000h, 010h, 00ch, 000h, 010h, 010h, 000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h
3538 00015EBD 00081000041000      <1>
3539 00015EC4 0010000010040010 08- <1>     db  000h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h
3539 00015ECD 00100C00101000      <1>
3540 00015ED4 0C10000810000410 08- <1>     db  00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h
3540 00015EDD 08100A08100C08      <1>
3541 00015EE4 100E08101008101008- <1>     db  010h, 00eh, 008h, 010h, 010h, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah
3541 00015EED 0E10080C10080A      <1>
3542 00015EF4 100808100A08100C08- <1>     db  010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh
3542 00015EFD 100E081010080E      <1>
3543 00015F04 10080C10080A100808- <1>     db  010h, 008h, 00ch, 010h, 008h, 00ah, 010h, 008h, 008h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h
3543 00015F0D 100808100A0810      <1>
3544 00015F14 0C08100E081010080E- <1>     db  00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah, 010h
3544 00015F1D 10080C10080A10      <1>
3545 00015F24 0B0B100C0B100D0B10- <1>     db  00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 010h, 010h
3545 00015F2D 0F0B10100B1010      <1>
3546 00015F34 0B0F100B0D100B0C10- <1>     db  00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh
3546 00015F3D 0B0B100C0B100D      <1>
3547 00015F44 0B100F0B10100B0F10- <1>     db  00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh
3547 00015F4D 0B0D100B0C100B      <1>
```

```
3548 00015F54 0B100B0B100C0B100D- <1>     db  00bh, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh
3548 00015F5D 0B100F0B10100B      <1>
3549 00015F64 0F100B0D100B0C1000- <1>     db  00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3549 00015F6D 00000000000000      <1>
3550 00015F74 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3550 00015F7D 00000000000000      <1>
3551                              <1>
3552                              <1>
3553                              <1> ; 04/07/2016
3554                              <1> ; FONT DATA
3555                              <1>
3556                              <1> CRT_CHAR_GEN:
3557                              <1> vgafont8:
3558 00015F84 0000000000000007E- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 081h, 0a5h, 081h, 0bdh, 099h, 081h, 07eh
3558 00015F8D 81A581BD99817E      <1>
3559 00015F94 7EFFDBFFC3E7FF7E6C- <1>     db  07eh, 0ffh, 0dbh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 06ch, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h
3559 00015F9D FEFEFE7C381000      <1>
3560 00015FA4 10387CFE7C38100038- <1>     db  010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 038h, 07ch, 038h, 0feh, 0feh, 07ch, 038h, 07ch
3560 00015FAD 7C38FEFE7C387C      <1>
3561 00015FB4 1010387CFE7C387C00- <1>     db  010h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 07ch, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h
3561 00015FBD 00183C3C180000      <1>
3562 00015FC4 FFFFE7C3C3E7FFFF00- <1>     db  0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h
3562 00015FCD 3C664242663C00      <1>
3563 00015FD4 FFC399BDBD99C3FF0F- <1>     db  0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 00fh, 007h, 00fh, 07dh, 0cch, 0cch, 0cch, 078h
3563 00015FDD 070F7DCCCCCC78      <1>
3564 00015FE4 3C6666663C187E183F- <1>     db  03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 03fh, 033h, 03fh, 030h, 030h, 070h, 0f0h, 0e0h
3564 00015FED 333F303070F0E0      <1>
3565 00015FF4 7F637F636367E6C099- <1>     db  07fh, 063h, 07fh, 063h, 063h, 067h, 0e6h, 0c0h, 099h, 05ah, 03ch, 0e7h, 0e7h, 03ch, 05ah, 099h
3565 00015FFD 5A3CE7E73C5A99      <1>
3566 00016004 80E0F8FEF8E0800002- <1>     db  080h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 080h, 000h, 002h, 00eh, 03eh, 0feh, 03eh, 00eh, 002h, 000h
3566 0001600D 0E3EFE3E0E0200      <1>
3567 00016014 183C7E18187E3C1866- <1>     db  018h, 03ch, 07eh, 018h, 018h, 07eh, 03ch, 018h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 000h
3567 0001601D 66666666006600      <1>
3568 00016024 7FDBDB7B1B1B1B003E- <1>     db  07fh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 000h, 03eh, 063h, 038h, 06ch, 06ch, 038h, 0cch, 078h
3568 0001602D 63386C6C38CC78      <1>
3569 00016034 000000007E7E7E0018- <1>     db  000h, 000h, 000h, 000h, 07eh, 07eh, 07eh, 000h, 018h, 03ch, 07eh, 018h, 07eh, 03ch, 018h, 0ffh
3569 0001603D 3C7E187E3C18FF      <1>
3570 00016044 183C7E181818180018- <1>     db  018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h
3570 0001604D 1818187E3C1800      <1>
3571 00016054 00180CFE0C18000000- <1>     db  000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h
3571 0001605D 3060FE60300000      <1>
3572 00016064 0000C0C0C0FE000000- <1>     db  000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h
3572 0001606D 2466FF66240000      <1>
3573 00016074 00183C7EFFFF000000- <1>     db  000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 000h, 000h, 000h, 0ffh, 0ffh, 07eh, 03ch, 018h, 000h, 000h
3573 0001607D FFFF7E3C180000      <1>
3574 00016084 000000000000000030- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 078h, 078h, 030h, 030h, 000h, 030h, 000h
3574 0001608D 78783030003000      <1>
3575 00016094 6C6C6C000000000006C- <1>     db  06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 0feh, 06ch, 06ch, 000h
3575 0001609D 6CFE6CFE6C6C00      <1>
3576 000160A4 307CC0780CF8300000- <1>     db  030h, 07ch, 0c0h, 078h, 00ch, 0f8h, 030h, 000h, 000h, 0c6h, 0cch, 018h, 030h, 066h, 0c6h, 000h
3576 000160AD C6CC183066C600      <1>
3577 000160B4 386C3876DCCC760060- <1>     db  038h, 06ch, 038h, 076h, 0dch, 0cch, 076h, 000h, 060h, 060h, 0c0h, 000h, 000h, 000h, 000h, 000h
3577 000160BD 60C000000000000     <1>
3578 000160C4 1830606060301800060- <1>     db  018h, 030h, 060h, 060h, 060h, 030h, 018h, 000h, 060h, 030h, 018h, 018h, 018h, 030h, 060h, 000h
3578 000160CD 30181818306000      <1>
3579 000160D4 00663CFF3C66000000- <1>     db  000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 000h
3579 000160DD 3030FC30300000      <1>
3580 000160E4 000000000030306000- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 060h, 000h, 000h, 000h, 0fch, 000h, 000h, 000h, 000h
3580 000160ED 0000FC00000000      <1>
3581 000160F4 000000000030300006- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 000h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h
3581 000160FD 0C183060C08000      <1>
3582 00016104 7CC6CEDEF6E67C0030- <1>     db  07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 07ch, 000h, 030h, 070h, 030h, 030h, 030h, 030h, 0fch, 000h
3582 0001610D 7030303030FC00      <1>
3583 00016114 78CC0C3860CCFC0078- <1>     db  078h, 0cch, 00ch, 038h, 060h, 0cch, 0fch, 000h, 078h, 0cch, 00ch, 038h, 00ch, 0cch, 078h, 000h
3583 0001611D CC0C380CCC7800      <1>
3584 00016124 1C3C6CCCFE0C1E00FC- <1>     db  01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 01eh, 000h, 0fch, 0c0h, 0f8h, 00ch, 00ch, 0cch, 078h, 000h
3584 0001612D C0F80C0CCC7800      <1>
3585 00016134 3860C0F8CCCC7800FC- <1>     db  038h, 060h, 0c0h, 0f8h, 0cch, 0cch, 078h, 000h, 0fch, 0cch, 00ch, 018h, 030h, 030h, 030h, 000h
3585 0001613D CC0C1830303000      <1>
3586 00016144 78CCCC78CCCC780078- <1>     db  078h, 0cch, 0cch, 078h, 0cch, 0cch, 078h, 000h, 078h, 0cch, 0cch, 07ch, 00ch, 018h, 070h, 000h
3586 0001614D CCCC7C0C187000      <1>
3587 00016154 0030300000303000000- <1>     db  000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 060h
3587 0001615D 30300000303060      <1>
3588 00016164 183060C06030180000- <1>     db  018h, 030h, 060h, 0c0h, 060h, 030h, 018h, 000h, 000h, 000h, 0fch, 000h, 000h, 0fch, 000h, 000h
3588 0001616D 00FC0000FC0000      <1>
3589 00016174 6030180C1830600078- <1>     db  060h, 030h, 018h, 00ch, 018h, 030h, 060h, 000h, 078h, 0cch, 00ch, 018h, 030h, 000h, 030h, 000h
3589 0001617D CC0C1830003000      <1>
3590 00016184 7CC6DEDEDEC0780030- <1>     db  07ch, 0c6h, 0deh, 0deh, 0deh, 0c0h, 078h, 000h, 030h, 078h, 0cch, 0cch, 0fch, 0cch, 0cch, 000h
3590 0001618D 78CCCCFCCCCC00      <1>
3591 00016194 FC66667C6666FC003C- <1>     db  0fch, 066h, 066h, 07ch, 066h, 066h, 0fch, 000h, 03ch, 066h, 0c0h, 0c0h, 0c0h, 066h, 03ch, 000h
3591 0001619D 66C0C0C0663C00      <1>
3592 000161A4 F86C6666666CF800FE- <1>     db  0f8h, 06ch, 066h, 066h, 066h, 06ch, 0f8h, 000h, 0feh, 062h, 068h, 078h, 068h, 062h, 0feh, 000h
3592 000161AD 6268786862FE00      <1>
3593 000161B4 FE6268786860F0003C- <1>     db  0feh, 062h, 068h, 078h, 068h, 060h, 0f0h, 000h, 03ch, 066h, 0c0h, 0c0h, 0ceh, 066h, 03eh, 000h
3593 000161BD 66C0C0CE663E00      <1>
3594 000161C4 CCCCCCFCCCCCCC0078- <1>     db  0cch, 0cch, 0cch, 0fch, 0cch, 0cch, 0cch, 000h, 078h, 030h, 030h, 030h, 030h, 030h, 078h, 000h
3594 000161CD 30303030307800      <1>
3595 000161D4 1E0C0C0CCCCC7800E6- <1>     db  01eh, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 0e6h, 066h, 06ch, 078h, 06ch, 066h, 0e6h, 000h
3595 000161DD 666C786C66E600      <1>
3596 000161E4 F06060606266FE00C6- <1>     db  0f0h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 000h
3596 000161ED EEFEFED6C6C600      <1>
3597 000161F4 C6E6F6DECEC6C60038- <1>     db  0c6h, 0e6h, 0f6h, 0deh, 0ceh, 0c6h, 0c6h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h
3597 000161FD 6CC6C6C66C3800      <1>
3598 00016204 FC66667C6060F00078- <1>     db  0fch, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 078h, 0cch, 0cch, 0cch, 0dch, 078h, 01ch, 000h
3598 0001620D CCCCCCDC781C00      <1>
3599 00016214 FC66667C6C66E60078- <1>     db  0fch, 066h, 066h, 07ch, 06ch, 066h, 0e6h, 000h, 078h, 0cch, 0e0h, 070h, 01ch, 0cch, 078h, 000h
3599 0001621D CCE0701CCC7800      <1>
3600 00016224 FCB4303030307800CC- <1>     db  0fch, 0b4h, 030h, 030h, 030h, 030h, 078h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 0fch, 000h
3600 0001622D CCCCCCCCCCFC00      <1>
3601 00016234 CCCCCCCCCC783000C6- <1>     db  0cch, 0cch, 0cch, 0cch, 0cch, 078h, 030h, 000h, 0c6h, 0c6h, 0c6h, 0d6h, 0feh, 0eeh, 0c6h, 000h
3601 0001623D C6C6D6FEEEC600      <1>
3602 00016244 C6C66C38386CC600CC- <1>     db  0c6h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 030h, 078h, 000h
3602 0001624D CCCC7830307800      <1>
3603 00016254 FEC68C183266FE0078- <1>     db  0feh, 0c6h, 08ch, 018h, 032h, 066h, 0feh, 000h, 078h, 060h, 060h, 060h, 060h, 060h, 078h, 000h
3603 0001625D 60606060607800      <1>
3604 00016264 C06030180C06020078- <1>     db  0c0h, 060h, 030h, 018h, 00ch, 006h, 002h, 000h, 078h, 018h, 018h, 018h, 018h, 018h, 078h, 000h
3604 0001626D 18181818187800      <1>
3605 00016274 10386CC600000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
3605 0001627D 000000000000FF      <1>
3606 00016284 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 076h, 000h
3606 0001628D 00780C7CCC7600      <1>
```

```
3607 00016294 E060607C6666DC0000- <1>     db  0e0h, 060h, 060h, 07ch, 066h, 066h, 0dch, 000h, 000h, 000h, 078h, 0cch, 0c0h, 0cch, 078h, 000h
3607 0001629D 0078CCC0CC7800      <1>
3608 000162A4 1C0C0C7CCCCC760000- <1>     db  01ch, 00ch, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
3608 000162AD 0078CCFCC07800      <1>
3609 000162B4 386C60F06060F00000- <1>     db  038h, 06ch, 060h, 0f0h, 060h, 060h, 0f0h, 000h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 0f8h
3609 000162BD 0076CCCC7C0CF8      <1>
3610 000162C4 E0606C766666E60030- <1>     db  0e0h, 060h, 06ch, 076h, 066h, 066h, 0e6h, 000h, 030h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3610 000162CD 00703030307800      <1>
3611 000162D4 0C000C0C0CCCCC78E0- <1>     db  00ch, 000h, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 0e0h, 060h, 066h, 06ch, 078h, 06ch, 0e6h, 000h
3611 000162DD 60666C786CE600      <1>
3612 000162E4 703030303030780000- <1>     db  070h, 030h, 030h, 030h, 030h, 030h, 078h, 000h, 000h, 000h, 0cch, 0feh, 0feh, 0d6h, 0c6h, 000h
3612 000162ED 00CCFEFED6C600      <1>
3613 000162F4 0000F8CCCCCCCC0000- <1>     db  000h, 000h, 0f8h, 0cch, 0cch, 0cch, 0cch, 000h, 000h, 000h, 078h, 0cch, 0cch, 0cch, 078h, 000h
3613 000162FD 0078CCCCCC7800      <1>
3614 00016304 0000DC66667C60F000- <1>     db  000h, 000h, 0dch, 066h, 066h, 07ch, 060h, 0f0h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 01eh
3614 0001630D 0076CCCC7C0C1E      <1>
3615 00016314 0000DC766660F00000- <1>     db  000h, 000h, 0dch, 076h, 066h, 060h, 0f0h, 000h, 000h, 000h, 07ch, 0c0h, 078h, 00ch, 0f8h, 000h
3615 0001631D 007CC0780CF800      <1>
3616 00016324 10307C303034180000- <1>     db  010h, 030h, 07ch, 030h, 030h, 034h, 018h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 076h, 000h
3616 0001632D 00CCCCCCCC7600      <1>
3617 00016334 0000CCCCCC78300000- <1>     db  000h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 000h, 000h, 000h, 0c6h, 0d6h, 0feh, 0feh, 06ch, 000h
3617 0001633D 00C6D6FEFE6C00      <1>
3618 00016344 0000C66C386CC60000- <1>     db  000h, 000h, 0c6h, 06ch, 038h, 06ch, 0c6h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 07ch, 00ch, 0f8h
3618 0001634D 00CCCCCC7C0CF8      <1>
3619 00016354 0000FC983064FC001C- <1>     db  000h, 000h, 0fch, 098h, 030h, 064h, 0fch, 000h, 01ch, 030h, 030h, 0e0h, 030h, 030h, 01ch, 000h
3619 0001635D 3030E030301C00      <1>
3620 00016364 18181800181818800E0- <1>     db  018h, 018h, 018h, 000h, 018h, 018h, 018h, 000h, 0e0h, 030h, 030h, 01ch, 030h, 030h, 0e0h, 000h
3620 0001636D 30301C3030E000      <1>
3621 00016374 76DC000000000000000- <1>     db  076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h
3621 0001637D 10386CC6C6FE00      <1>
3622 00016384 78CCC0CC78180C7800- <1>     db  078h, 0cch, 0c0h, 0cch, 078h, 018h, 00ch, 078h, 000h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
3622 0001638D CC00CCCCCC7E00      <1>
3623 00016394 1C0078CCFCC078007E- <1>     db  01ch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 07eh, 0c3h, 03ch, 006h, 03eh, 066h, 03fh, 000h
3623 0001639D C33C063E663F00      <1>
3624 000163A4 CC00780C7CCC7E00E0- <1>     db  0cch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 0e0h, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h
3624 000163AD 00780C7CCC7E00      <1>
3625 000163B4 3030780C7CCC7E0000- <1>     db  030h, 030h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 000h, 000h, 078h, 0c0h, 0c0h, 078h, 00ch, 038h
3625 000163BD 0078C0C0780C38      <1>
3626 000163C4 7EC33C667E603C00CC- <1>     db  07eh, 0c3h, 03ch, 066h, 07eh, 060h, 03ch, 000h, 0cch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
3626 000163CD 0078CCFCC07800      <1>
3627 000163D4 E00078CCFCC07800CC- <1>     db  0e0h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 0cch, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3627 000163DD 00703030307800      <1>
3628 000163E4 7CC6381818183C00E0- <1>     db  07ch, 0c6h, 038h, 018h, 018h, 018h, 03ch, 000h, 0e0h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3628 000163ED 00703030307800      <1>
3629 000163F4 C6386CC6FEC6C60030- <1>     db  0c6h, 038h, 06ch, 0c6h, 0feh, 0c6h, 0c6h, 000h, 030h, 030h, 000h, 078h, 0cch, 0fch, 0cch, 000h
3629 000163FD 300078CCFCCC00      <1>
3630 00016404 1C00FC607860FC0000- <1>     db  01ch, 000h, 0fch, 060h, 078h, 060h, 0fch, 000h, 000h, 000h, 07fh, 00ch, 07fh, 0cch, 07fh, 000h
3630 0001640D 007F0C7FCC7F00      <1>
3631 00016414 3E6CCCFECCCCCE0078- <1>     db  03eh, 06ch, 0cch, 0feh, 0cch, 0cch, 0ceh, 000h, 078h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h
3631 0001641D CC0078CCCC7800      <1>
3632 00016424 00CC0078CCCC780000- <1>     db  000h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 0e0h, 000h, 078h, 0cch, 0cch, 078h, 000h
3632 0001642D E00078CCCC7800      <1>
3633 00016434 78CC00CCCCCC7E0000- <1>     db  078h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h, 000h, 0e0h, 000h, 0cch, 0cch, 0cch, 07eh, 000h
3633 0001643D E000CCCCCC7E00      <1>
3634 00016444 00CC00CCCC7C0CF8C3- <1>     db  000h, 0cch, 000h, 0cch, 0cch, 07ch, 00ch, 0f8h, 0c3h, 018h, 03ch, 066h, 066h, 03ch, 018h, 000h
3634 0001644D 183C66663C1800      <1>
3635 00016454 CC00CCCCCCCC780018- <1>     db  0cch, 000h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 018h, 018h, 07eh, 0c0h, 0c0h, 07eh, 018h, 018h
3635 0001645D 187EC0C07E1818      <1>
3636 00016464 386C64F060E6FC00CC- <1>     db  038h, 06ch, 064h, 0f0h, 060h, 0e6h, 0fch, 000h, 0cch, 0cch, 078h, 0fch, 030h, 0fch, 030h, 030h
3636 0001646D CC78FC30FC3030      <1>
3637 00016474 F8CCCCFAC6CFC6C70E- <1>     db  0f8h, 0cch, 0cch, 0fah, 0c6h, 0cfh, 0c6h, 0c7h, 00eh, 01bh, 018h, 03ch, 018h, 018h, 0d8h, 070h
3637 0001647D 1B183C1818D870      <1>
3638 00016484 1C00780C7CCC7E0038- <1>     db  01ch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 038h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3638 0001648D 00703030307800      <1>
3639 00016494 001C0078CCCC780000- <1>     db  000h, 01ch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 01ch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
3639 0001649D 1C00CCCCCC7E00      <1>
3640 000164A4 00F800F8CCCCCC00FC- <1>     db  000h, 0f8h, 000h, 0f8h, 0cch, 0cch, 0cch, 000h, 0fch, 000h, 0cch, 0ech, 0fch, 0dch, 0cch, 000h
3640 000164AD 00CCECFCDCCC00      <1>
3641 000164B4 3C6C6C3E007E000038- <1>     db  03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h
3641 000164BD 6C6C38007C0000      <1>
3642 000164C4 30003060C0CC780000- <1>     db  030h, 000h, 030h, 060h, 0c0h, 0cch, 078h, 000h, 000h, 000h, 000h, 0fch, 0c0h, 0c0h, 000h, 000h
3642 000164CD 0000FCC0C00000      <1>
3643 000164D4 000000FC0C0C0000C3- <1>     db  000h, 000h, 000h, 0fch, 00ch, 00ch, 000h, 000h, 0c3h, 0c6h, 0cch, 0deh, 033h, 066h, 0cch, 00fh
3643 000164DD C6CCDE3366CC0F      <1>
3644 000164E4 C3C6CCDB376FCF0318- <1>     db  0c3h, 0c6h, 0cch, 0dbh, 037h, 06fh, 0cfh, 003h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 000h
3644 000164ED 18001818181800      <1>
3645 000164F4 003366CC6633000000- <1>     db  000h, 033h, 066h, 0cch, 066h, 033h, 000h, 000h, 000h, 0cch, 066h, 033h, 066h, 0cch, 000h, 000h
3645 000164FD CC663366CC0000      <1>
3646 00016504 228822882288228855- <1>     db  022h, 088h, 022h, 088h, 022h, 088h, 022h, 088h, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
3646 0001650D AA55AA55AA55AA      <1>
3647 00016514 DB77DBEEDB77DBEE18- <1>     db  0dbh, 077h, 0dbh, 0eeh, 0dbh, 077h, 0dbh, 0eeh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
3647 0001651D 18181818181818      <1>
3648 00016524 18181818F818181818- <1>     db  018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h
3648 0001652D 18F818F8181818      <1>
3649 00016534 36363636F636363600- <1>     db  036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h
3649 0001653D 000000FE363636      <1>
3650 00016544 0000F818F818181836- <1>     db  000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h
3650 0001654D 36F606F6363636      <1>
3651 00016554 3636363636363600- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h
3651 0001655D 00FE06F6363636      <1>
3652 00016564 3636F606FE00000036- <1>     db  036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h
3652 0001656D 363636FE000000      <1>
3653 00016574 1818F818F800000000- <1>     db  018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h
3653 0001657D 000000F8181818      <1>
3654 00016584 181818181F00000018- <1>     db  018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h
3654 0001658D 181818FF000000      <1>
3655 00016594 00000000FF18181818- <1>     db  000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h
3655 0001659D 1818181F181818      <1>
3656 000165A4 00000000FF00000018- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h
3656 000165AD 181818FF181818      <1>
3657 000165B4 18181F181F18181836- <1>     db  018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h
3657 000165BD 36363637363636      <1>
3658 000165C4 363637303F00000000- <1>     db  036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h
3658 000165CD 003F3037363636      <1>
3659 000165D4 3636F700FF00000000- <1>     db  036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h
3659 000165DD 00FF00F7363636      <1>
3660 000165E4 3636373037363636 00- <1>     db  036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
3660 000165ED 00FF00FF000000      <1>
3661 000165F4 3636F700F736363618- <1>     db  036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
3661 000165FD 18FF00FF000000      <1>
3662 00016604 36363636FF00000000- <1>     db  036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h
```

```
3662 0001660D 00FF00FF181818      <1>
3663 00016614 00000000FF36363636- <1>     db  000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h
3663 0001661D 3636363F000000      <1>
3664 00016624 18181F181F00000000- <1>     db  018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h
3664 0001662D 001F181F181818      <1>
3665 00016634 000000003F36363636- <1>     db  000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h
3665 0001663D 363636FF363636      <1>
3666 00016644 1818FF18FF18181818- <1>     db  018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h
3666 0001664D 181818F8000000      <1>
3667 00016654 000000001F181818FF- <1>     db  000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3667 0001665D FFFFFFFFFFFFFF      <1>
3668 00016664 00000000FFFFFFFFF0- <1>     db  000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
3668 0001666D F0F0F0F0F0F0F0      <1>
3669 00016674 0F0F0F0F0F0F0F0FFF- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h
3669 0001667D FFFFFF00000000      <1>
3670 00016684 000076DCC8DC760000- <1>     db  000h, 000h, 076h, 0dch, 0c8h, 0dch, 076h, 000h, 000h, 078h, 0cch, 0f8h, 0cch, 0f8h, 0c0h, 0c0h
3670 0001668D 78CCF8CCF8C0C0      <1>
3671 00016694 00FCCCC0C0C0C00000- <1>     db  000h, 0fch, 0cch, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
3671 0001669D FE6C6C6C6C6C00      <1>
3672 000166A4 FCCC603060CCFC0000- <1>     db  0fch, 0cch, 060h, 030h, 060h, 0cch, 0fch, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 070h, 000h
3672 000166AD 007ED8D8D87000      <1>
3673 000166B4 00666666667C60C000- <1>     db  000h, 066h, 066h, 066h, 066h, 07ch, 060h, 0c0h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 000h
3673 000166BD 76DC1818181800      <1>
3674 000166C4 FC3078CCCC7830FC38- <1>     db  0fch, 030h, 078h, 0cch, 0cch, 078h, 030h, 0fch, 038h, 06ch, 0c6h, 0feh, 0c6h, 06ch, 038h, 000h
3674 000166CD 6CC6FEC66C3800      <1>
3675 000166D4 386CC6C66C6CEE001C- <1>     db  038h, 06ch, 0c6h, 0c6h, 06ch, 06ch, 0eeh, 000h, 01ch, 030h, 018h, 07ch, 0cch, 0cch, 078h, 000h
3675 000166DD 30187CCCCC7800      <1>
3676 000166E4 00007EDBDB7E000006- <1>     db  000h, 000h, 07eh, 0dbh, 0dbh, 07eh, 000h, 000h, 006h, 00ch, 07eh, 0dbh, 0dbh, 07eh, 060h, 0c0h
3676 000166ED 0C7EDBDB7E60C0      <1>
3677 000166F4 3860C0F8C060380078- <1>     db  038h, 060h, 0c0h, 0f8h, 0c0h, 060h, 038h, 000h, 078h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 000h
3677 000166FD CCCCCCCCCCCC00      <1>
3678 00016704 00FC00FC00FC000030- <1>     db  000h, 0fch, 000h, 0fch, 000h, 0fch, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 0fch, 000h
3678 0001670D 30FC303000FC00      <1>
3679 00016714 603018306000FC0018- <1>     db  060h, 030h, 018h, 030h, 060h, 000h, 0fch, 000h, 018h, 030h, 060h, 030h, 018h, 000h, 0fch, 000h
3679 0001671D 3060301800FC00      <1>
3680 00016724 0E1B1B181818181818- <1>     db  00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 070h
3680 0001672D 18181818D8D870      <1>
3681 00016734 303000FC0030300000- <1>     db  030h, 030h, 000h, 0fch, 000h, 030h, 030h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h
3681 0001673D 76DC0076DC0000      <1>
3682 00016744 386C6C3800000000000- <1>    db  038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h
3682 0001674D 00001818000000      <1>
3683 00016754 000000001800000000F- <1>    db  000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 0ech, 06ch, 03ch, 01ch
3683 0001675D 0C0C0CEC6C3C1C      <1>
3684 00016764 786C6C6C6C00000070- <1>     db  078h, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 070h, 018h, 030h, 060h, 078h, 000h, 000h, 000h
3684 0001676D 18306078000000      <1>
3685 00016774 00003C3C3C3C000000- <1>     db  000h, 000h, 03ch, 03ch, 03ch, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3685 0001677D 000000000000000     <1>
3686                              <1> vgafont14:
3687 00016784 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3687 0001678D 00000000000000      <1>
3688 00016794 7E81A58181BD99817E- <1>     db  07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 07eh, 000h, 000h, 000h, 000h, 000h, 07eh, 0ffh
3688 0001679D 00000000007EFF      <1>
3689 000167A4 DBFFFFC3E7FF7E0000- <1>     db  0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 0feh, 0feh
3689 000167AD 000000006CFEFE      <1>
3690 000167B4 FEFE7C3810000000000- <1>    db  0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch
3690 000167BD 000010387CFE7C      <1>
3691 000167C4 381000000000000018- <1>     db  038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h
3691 000167CD 3C3CE7E7E71818      <1>
3692 000167D4 3C00000000000183C7E- <1>    db  03ch, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h
3692 000167DD FFFF7E18183C00      <1>
3693 000167E4 0000000000000183C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
3693 000167ED 3C180000000000      <1>
3694 000167F4 FFFFFFFFFFE7C3C3E7- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h
3694 000167FD FFFFFFFFFF0000      <1>
3695 00016804 00003C664242663C00- <1>     db  000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh
3695 0001680D 000000FFFFFFFF      <1>
3696 00016814 C399BDBD99C3FFFFFF- <1>     db  0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 01eh, 00eh, 01ah, 032h
3696 0001681D FF00001E0E1A32      <1>
3697 00016824 78CCCCCC7800000000- <1>     db  078h, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 066h, 066h, 03ch, 018h
3697 0001682D 003C6666663C18      <1>
3698 00016834 7E1818000000000003F- <1>    db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 070h, 0f0h
3698 0001683D 333F30303070F0      <1>
3699 00016844 E000000000007F637F- <1>     db  0e0h, 000h, 000h, 000h, 000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h
3699 0001684D 63636367E7E6C0      <1>
3700 00016854 000000001818DB3CE7- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h
3700 0001685D 3CDB1818000000      <1>
3701 00016864 000080C0E0F8FEF8E0- <1>     db  000h, 000h, 080h, 0c0h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h, 000h
3701 0001686D C0800000000000      <1>
3702 00016874 02060E3EFE3E0E0602- <1>     db  002h, 006h, 00eh, 03eh, 0feh, 03eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch
3702 0001687D 0000000000183C      <1>
3703 00016884 7E1818187E3C180000- <1>     db  07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h
3703 0001688D 00000066666666      <1>
3704 00016894 666600666600000000- <1>     db  066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh
3704 0001689D 007FDBDBDB7B1B      <1>
3705 000168A4 1B1B1B000000007CC6- <1>     db  01bh, 01bh, 01bh, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h
3705 000168AD 60386CC6C66C38      <1>
3706 000168B4 0CC67C0000000000000- <1>    db  00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 000h
3706 000168BD 000000FEFEFE00      <1>
3707 000168C4 00000000183C7E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h
3707 000168CD 187E3C187E0000      <1>
3708 000168D4 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
3708 000168DD 18180000000000      <1>
3709 000168E4 1818181818187E3C18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3709 000168ED 00000000000000      <1>
3710 000168F4 180CFE0C1800000000- <1>     db  018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 060h
3710 000168FD 00000000003060      <1>
3711 00016904 FE60300000000000000- <1>    db  0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h
3711 0001690D 00000000C0C0C0      <1>
3712 00016914 FE00000000000000000- <1>    db  0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 028h, 06ch, 0feh, 06ch, 028h, 000h
3712 0001691D 00286CFE6C2800      <1>
3713 00016924 000000000000001038- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h
3713 0001692D 387C7CFEFE0000      <1>
3714 00016934 0000000000FEFE7C7C- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h
3714 0001693D 38381000000000      <1>
3715 00016944 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3715 0001694D 00000000000000      <1>
3716 00016954 183C3C3C1818001818- <1>     db  018h, 03ch, 03ch, 03ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 066h, 066h, 066h
3716 0001695D 00000000666666      <1>
3717 00016964 240000000000000000- <1>     db  024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch
3717 0001696D 0000006C6CFE6C      <1>
3718 00016974 6C6CFE6C6C00000018- <1>     db  06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h
```

```
3718 0001697D 187CC6C2C07C06      <1>
3719 00016984 86C67C181800000000- <1>     db  086h, 0c6h, 07ch, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 066h
3719 0001698D 00C2C60C183066      <1>
3720 00016994 C600000000000386C6C- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 076h, 000h
3720 0001699D 3876DCCCCC7600      <1>
3721 000169A4 000000303030600000- <1>     db  000h, 000h, 000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3721 000169AD 00000000000000      <1>
3722 000169B4 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h, 000h
3722 000169BD 180C0000000000      <1>
3723 000169C4 30180C0C0C0C0C1830- <1>     db  030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3723 000169CD 00000000000000      <1>
3724 000169D4 663CFF3C6600000000- <1>     db  066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
3724 000169DD 00000000001818      <1>
3725 000169E4 7E1818000000000000- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3725 000169ED 00000000000000      <1>
3726 000169F4 1818183000000000000- <1>     db  018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h
3726 000169FD 000000FE000000      <1>
3727 00016A04 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
3727 00016A0D 00000000181800      <1>
3728 00016A14 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
3728 00016A1D 60C08000000000      <1>
3729 00016A24 00007CC6CEDEF6E6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3729 00016A2D C67C0000000000      <1>
3730 00016A34 183878181818187E- <1>     db  018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h
3730 00016A3D 00000000007CC6      <1>
3731 00016A44 060C183060C6FE0000- <1>     db  006h, 00ch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 006h, 006h
3731 00016A4D 0000007CC60606      <1>
3732 00016A54 3C0606C67C00000000- <1>     db  03ch, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh
3732 00016A5D 000C1C3C6CCCFE      <1>
3733 00016A64 0C0C1E00000000000FE- <1>     db  00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 0c6h
3733 00016A6D C0C0C0FC0606C6      <1>
3734 00016A74 7C00000000003860C0- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 07ch, 000h
3734 00016A7D C0FCC6C6C67C00      <1>
3735 00016A84 00000000FEC6060C18- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c6h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h
3735 00016A8D 303030300000000     <1>
3736 00016A94 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3736 00016A9D C67C0000000000      <1>
3737 00016AA4 7CC6C6C67E06060C78- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h, 000h, 000h, 018h
3737 00016AAD 00000000000018      <1>
3738 00016AB4 1800000018180000000- <1>     db  018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
3738 00016ABD 00000000181800      <1>
3739 00016AC4 0000181830000000000- <1>     db  000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h
3739 00016ACD 00060C18306030      <1>
3740 00016AD4 180C060000000000000- <1>     db  018h, 00ch, 006h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h
3740 00016ADD 00007E00007E00      <1>
3741 00016AE4 0000000000000603018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h
3741 00016AED 0C060C18306000      <1>
3742 00016AF4 000000007CC6C60C18- <1>     db  000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h
3742 00016AFD 18001818000000      <1>
3743 00016B04 00007CC6C6DEDEDEDC- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h, 000h
3743 00016B0D C07C0000000000      <1>
3744 00016B14 10386CC6C6FEC6C6C6- <1>     db  010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h, 0fch, 066h
3744 00016B1D 00000000FC66      <1>
3745 00016B24 66667C666666FC0000- <1>     db  066h, 066h, 07ch, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h
3745 00016B2D 0000003C66C2C0      <1>
3746 00016B34 C0C0C2663C00000000- <1>     db  0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h
3746 00016B3D 00F86C66666666      <1>
3747 00016B44 666CF800000000000FE- <1>     db  066h, 06ch, 0f8h, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 062h, 066h
3747 00016B4D 66626878686266      <1>
3748 00016B54 FE0000000000FE6662- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 0f0h, 000h
3748 00016B5D 6878686060F000      <1>
3749 00016B64 000000003C66C2C0C0- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 066h, 03ah, 000h, 000h, 000h
3749 00016B6D DEC6663A000000      <1>
3750 00016B74 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
3750 00016B7D C6C600000000000     <1>
3751 00016B84 3C1818181818183C- <1>     db  03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 01eh, 00ch
3751 00016B8D 00000000001E0C      <1>
3752 00016B94 0C0C0C0CCCCC780000- <1>     db  00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 0e6h, 066h, 06ch, 06ch
3752 00016B9D 000000E6666C6C      <1>
3753 00016BA4 786C6C66E600000000- <1>     db  078h, 06ch, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h
3753 00016BAD 00F06060606060      <1>
3754 00016BB4 6266FE00000000000C6- <1>     db  062h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 0c6h
3754 00016BBD EEFEFED6C6C6C6      <1>
3755 00016BC4 C60000000000C6E6F6- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h
3755 00016BCD FEDECEC6C6C600      <1>
3756 00016BD4 000000000386CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h
3756 00016BDD C6C66C38000000      <1>
3757 00016BE4 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h
3757 00016BED 60F00000000000      <1>
3758 00016BF4 7CC6C6C6C6D6DE7C0C- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h, 000h, 000h, 0fch, 066h
3758 00016BFD 0E00000000FC66      <1>
3759 00016C04 66667C6C6666E60000- <1>     db  066h, 066h, 07ch, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 060h
3759 00016C0D 0000007CC6C660      <1>
3760 00016C14 380CC6C67C00000000- <1>     db  038h, 00ch, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 07eh, 07eh, 05ah, 018h, 018h, 018h
3760 00016C1D 007E7E5A181818      <1>
3761 00016C24 18183C00000000000C6- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h
3761 00016C2D C6C6C6C6C6C6C6      <1>
3762 00016C34 7C0000000000C6C6C6- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 010h, 000h
3762 00016C3D C6C66C381000      <1>
3763 00016C44 00000000C6C6C6C6D6- <1>     db  000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 07ch, 06ch, 000h, 000h, 000h
3763 00016C4D D6FE7C6C000000      <1>
3764 00016C54 0000C6C66C3838386C- <1>     db  000h, 000h, 0c6h, 0c6h, 06ch, 038h, 038h, 038h, 06ch, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
3764 00016C5D C6C600000000000     <1>
3765 00016C64 666666663C1818183C- <1>     db  066h, 066h, 066h, 066h, 03ch, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h
3765 00016C6D 0000000000FEC6      <1>
3766 00016C74 8C183060C2C6FE0000- <1>     db  08ch, 018h, 030h, 060h, 0c2h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 03ch, 030h, 030h, 030h
3766 00016C7D 0000003C303030      <1>
3767 00016C84 303030303C00000000- <1>     db  030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch
3767 00016C8D 0080C0E070381C      <1>
3768 00016C94 0E0602000000000003C- <1>     db  00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch
3768 00016C9D 0C0C0C0C0C0C0C      <1>
3769 00016CA4 3C00000010386CC600- <1>     db  03ch, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3769 00016CAD 00000000000000      <1>
3770 00016CB4 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h
3770 00016CBD 0000000000FF00      <1>
3771 00016CC4 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3771 00016CCD 00000000000000      <1>
3772 00016CD4 000000780C7CCCCC76- <1>     db  000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0e0h, 060h
3772 00016CDD 0000000000E060      <1>
3773 00016CE4 60786C6666667C0000- <1>     db  060h, 078h, 06ch, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
3773 00016CED 00000000000007C     <1>
```

```
3774 00016CF4 C6C0C0C67C00000000- <1>     db  0c6h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch
3774 00016CFD 001C0C0C3C6CCC      <1>
3775 00016D04 CCCC760000000000- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h
3775 00016D0D 00007CC6FEC0C6      <1>
3776 00016D14 7C0000000000386C64- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 0f0h, 000h
3776 00016D1D 60F0606060F000      <1>
3777 00016D24 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
3777 00016D2D CCCC7C0CCC7800      <1>
3778 00016D34 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h
3778 00016D3D 66E60000000000      <1>
3779 00016D44 181800381818183C- <1>     db  018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 006h, 006h
3779 00016D4D 00000000000606      <1>
3780 00016D54 000E0606060666663C- <1>     db  000h, 00eh, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h, 000h, 000h, 0e0h, 060h, 060h, 066h
3780 00016D5D 000000E0606066      <1>
3781 00016D64 6C786C66E600000000- <1>     db  06ch, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h
3781 00016D6D 00381818181818      <1>
3782 00016D74 18183C0000000000000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ech, 0feh, 0d6h, 0d6h, 0d6h
3782 00016D7D 0000ECFED6D6D6      <1>
3783 00016D84 C600000000000000000- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 000h
3783 00016D8D DC666666666600      <1>
3784 00016D94 000000000000007CC6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h
3784 00016D9D C6C6C67C000000      <1>
3785 00016DA4 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 000h, 000h
3785 00016DAD 7C6060F0000000      <1>
3786 00016DB4 00000076CCCCCC7C0C- <1>     db  000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h
3786 00016DBD 0C1E0000000000      <1>
3787 00016DC4 00DC76666060F00000- <1>     db  000h, 0dch, 076h, 066h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
3787 00016DCD 0000000000007C      <1>
3788 00016DD4 C6701CC67C00000000- <1>     db  0c6h, 070h, 01ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h
3788 00016DDD 00103030FC3030      <1>
3789 00016DE4 30361C0000000000000- <1>     db  030h, 036h, 01ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch
3789 00016DED 0000CCCCCCCCCC      <1>
3790 00016DF4 760000000000000000- <1>     db  076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 03ch, 018h, 000h
3790 00016DFD 666666663C1800      <1>
3791 00016E04 000000000000000C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 06ch, 000h, 000h, 000h
3791 00016E0D D6D6FE6C000000      <1>
3792 00016E14 0000000000C66C3838- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h
3792 00016E1D 6CC60000000000      <1>
3793 00016E24 000000C6C6C6C67E06- <1>     db  000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h, 000h, 000h, 000h, 000h
3793 00016E2D 0CF80000000000      <1>
3794 00016E34 00FECC183066FE0000- <1>     db  000h, 0feh, 0cch, 018h, 030h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 00eh, 018h, 018h, 018h
3794 00016E3D 0000000E181818      <1>
3795 00016E44 701818180E00000000- <1>     db  070h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h
3795 00016E4D 00181818180018      <1>
3796 00016E54 1818180000000000070- <1>     db  018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h
3796 00016E5D 1818180E181818      <1>
3797 00016E64 700000000000076DC00- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3797 00016E6D 00000000000000      <1>
3798 00016E74 00000000000010386C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h
3798 00016E7D C6C6FE00000000      <1>
3799 00016E84 00003C66C2C0C0C266- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h, 000h
3799 00016E8D 3C0C067C000000      <1>
3800 00016E94 CCCC00CCCCCCCCCC76- <1>     db  0cch, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h
3800 00016E9D 000000000C1830      <1>
3801 00016EA4 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 078h
3801 00016EAD 000010386C0078      <1>
3802 00016EB4 0C7CCCCC7600000000- <1>     db  00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 000h, 078h, 00ch, 07ch
3802 00016EBD 00CCCC00780C7C      <1>
3803 00016EC4 CCCC76000000006030- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch
3803 00016ECD 1800780C7CCCCC      <1>
3804 00016ED4 7600000000386C3800- <1>     db  076h, 000h, 000h, 000h, 000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h
3804 00016EDD 780C7CCCCC7600      <1>
3805 00016EE4 0000000000003C6660- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h
3805 00016EED 663C0C063C0000      <1>
3806 00016EF4 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3806 00016EFD C67C0000000000      <1>
3807 00016F04 CCCC007CC6FEC0C67C- <1>     db  0cch, 0cch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h
3807 00016F0D 00000000603018      <1>
3808 00016F14 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 000h, 038h
3808 00016F1D 00000066660038      <1>
3809 00016F24 181818183C00000000- <1>     db  018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h
3809 00016F2D 183C6600381818      <1>
3810 00016F34 18183C000000006030- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h
3810 00016F3D 18003818181818      <1>
3811 00016F44 3C00000000C6C61038- <1>     db  03ch, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h
3811 00016F4D 6CC6C6FEC6C600      <1>
3812 00016F54 0000386C3800386CC6- <1>     db  000h, 000h, 038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h, 000h, 000h
3812 00016F5D C6FEC6C6000000      <1>
3813 00016F64 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h
3813 00016F6D 66FE0000000000      <1>
3814 00016F74 0000CC76367ED8D86E- <1>     db  000h, 000h, 0cch, 076h, 036h, 07eh, 0d8h, 0d8h, 06eh, 000h, 000h, 000h, 000h, 000h, 03eh, 06ch
3814 00016F7D 00000000003E6C      <1>
3815 00016F84 CCCCFECCCCCCCE0000- <1>     db  0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 07ch
3815 00016F8D 000010386C007C      <1>
3816 00016F94 C6C6C6C67C00000000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h, 07ch, 0c6h, 0c6h
3816 00016F9D 00C6C6007CC6C6      <1>
3817 00016FA4 C6C67C000000006030- <1>     db  0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h
3817 00016FAD 18007CC6C6C6C6      <1>
3818 00016FB4 7C000000003078CC00- <1>     db  07ch, 000h, 000h, 000h, 000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h
3818 00016FBD CCCCCCCCCC7600      <1>
3819 00016FC4 00000060301800CCCC- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h
3819 00016FCD CCCCCC76000000      <1>
3820 00016FD4 0000C6C600C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h, 000h, 0c6h
3820 00016FDD 7E060C780000C6      <1>
3821 00016FE4 C6386CC6C6C6C66C38- <1>     db  0c6h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h
3821 00016FED 00000000C6C600      <1>
3822 00016FF4 C6C6C6C6C6C67C0000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 018h, 03ch, 066h, 060h
3822 00016FFD 000018183C6660      <1>
3823 00017004 60663C1818000000000- <1>     db  060h, 066h, 03ch, 018h, 018h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h
3823 0001700D 386C6460F06060      <1>
3824 00017014 60E6FC000000000066- <1>     db  060h, 0e6h, 0fch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 03ch, 018h, 07eh, 018h, 07eh, 018h
3824 0001701D 663C187E187E18      <1>
3825 00017024 1800000000F8CCCCF8- <1>     db  018h, 000h, 000h, 000h, 000h, 0f8h, 0cch, 0cch, 0f8h, 0c4h, 0cch, 0deh, 0cch, 0cch, 0c6h, 000h
3825 0001702D C4CCDECCCCC600      <1>
3826 00017034 0000000E1B1818187E- <1>     db  000h, 000h, 000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h
3826 0001703D 18181818D87000      <1>
3827 00017044 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch
3827 0001704D CC760000000000      <1>
3828 00017054 183000381818183C- <1>     db  018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h
3828 0001705D 00000000183060      <1>
3829 00017064 007CC6C6C6C67C0000- <1>     db  000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h, 000h, 0cch
```

```
3829 0001706D 000018306000CC      <1>
3830 00017074 CCCCCCCC7600000000- <1>     db  0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h
3830 0001707D 0076DC00DC6666      <1>
3831 00017084 66666600000076DC00- <1>     db  066h, 066h, 066h, 000h, 000h, 000h, 076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h
3831 0001708D C6E6F6FEDECEC6      <1>
3832 00017094 C6000000003C6C6C3E- <1>     db  0c6h, 000h, 000h, 000h, 000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h
3832 0001709D 007E0000000000      <1>
3833 000170A4 000000386C6C38007C- <1>     db  000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3833 000170AD 00000000000000      <1>
3834 000170B4 0000303000303060C6- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3834 000170BD C67C0000000000      <1>
3835 000170C4 00000000FEC0C0C000- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3835 000170CD 00000000000000      <1>
3836 000170D4 0000FE060606000000- <1>     db  000h, 000h, 0feh, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h
3836 000170DD 0000C0C0C6CCD8      <1>
3837 000170E4 3060DC860C183E0000- <1>     db  030h, 060h, 0dch, 086h, 00ch, 018h, 03eh, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h, 030h, 066h
3837 000170ED C0C0C6CCD83066      <1>
3838 000170F4 CE9E3E0060600000018- <1>     db  0ceh, 09eh, 03eh, 006h, 006h, 000h, 000h, 000h, 018h, 018h, 000h, 018h, 018h, 03ch, 03ch, 03ch
3838 000170FD 180018183C3C3C      <1>
3839 00017104 180000000000000036- <1>     db  018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h
3839 0001710D 6CD86C36000000      <1>
3840 00017114 000000000000D86C36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h
3840 0001711D 6CD80000000000      <1>
3841 00017124 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 055h, 0aah
3841 0001712D 441144114455AA      <1>
3842 00017134 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 0ddh, 077h, 0ddh, 077h
3842 0001713D AA55AADD77DD77      <1>
3843 00017144 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 018h, 018h, 018h, 018h, 018h, 018h
3843 0001714D 77181818181818      <1>
3844 00017154 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h
3844 0001715D 181818181818F8      <1>
3845 00017164 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h
3845 0001716D 1818F818F81818      <1>
3846 00017174 1818181836363636- <1>     db  018h, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h
3846 0001717D 3636F636363636      <1>
3847 00017184 3636000000000000- <1>     db  036h, 036h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h
3847 0001718D FE363636363636      <1>
3848 00017194 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 036h, 036h
3848 0001719D 18181818183636      <1>
3849 000171A4 363636F606F6363636- <1>     db  036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
3849 000171AD 36363636363636      <1>
3850 000171B4 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0feh
3850 000171BD 3600000000000FE     <1>
3851 000171C4 06F6363636363636- <1>     db  006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh
3851 000171CD 36363636F606FE      <1>
3852 000171D4 0000000000000363636- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h
3852 000171DD 36363636FE0000      <1>
3853 000171E4 0000000018181818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h
3853 000171ED F818F800000000      <1>
3854 000171F4 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h
3854 000171FD F8181818181818      <1>
3855 00017204 181818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
3855 0001720D 00000000001818      <1>
3856 00017214 1818181818FF000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3856 0001721D 00000000000000      <1>
3857 00017224 000000FF1818181818- <1>     db  000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
3857 0001722D 18181818181818      <1>
3858 00017234 181F18181818181800- <1>     db  018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
3858 0001723D 0000000000000FF     <1>
3859 00017244 0000000000000181818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h
3859 0001724D 18181818FF1818      <1>
3860 00017254 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h
3860 0001725D 1F181F18181818      <1>
3861 00017264 1818363636363636- <1>     db  018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h
3861 0001726D 37363636363636      <1>
3862 00017274 3636363636373030F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3862 0001727D 00000000000000      <1>
3863 00017284 0000003F3037363636- <1>     db  000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
3863 0001728D 36363636363636      <1>
3864 00017294 36F700FF0000000000- <1>     db  036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
3864 0001729D 0000000000000FF     <1>
3865 000172A4 00F7363636363636- <1>     db  000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h
3865 000172AD 36363636373037      <1>
3866 000172B4 3636363636000000- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h
3866 000172BD 0000FF00FF0000      <1>
3867 000172C4 0000000036363636- <1>     db  000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h
3867 000172CD F700F736363636      <1>
3868 000172D4 363618181818FF00- <1>     db  036h, 036h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h
3868 000172DD FF0000000000000     <1>
3869 000172E4 3636363636636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3869 000172ED 00000000000000      <1>
3870 000172F4 000000FF00FF181818- <1>     db  000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
3870 000172FD 18181800000000      <1>
3871 00017304 000000FF3636363636- <1>     db  000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
3871 0001730D 36363636363636      <1>
3872 00017314 363F000000000000018- <1>     db  036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh
3872 0001731D 181818181F181F      <1>
3873 00017324 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h
3873 0001732D 00001F181F1818      <1>
3874 00017334 1818181800000000000- <1>     db  018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h
3874 0001733D 00003F36363636      <1>
3875 00017344 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h
3875 0001734D FF3636363636      <1>
3876 00017354 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
3876 0001735D 18181818181818      <1>
3877 00017364 1818181818F8000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3877 0001736D 00000000000000      <1>
3878 00017374 0000001F1818181818- <1>     db  000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3878 0001737D 18FFFFFFFFFFFF      <1>
3879 00017384 FFFFFFFFFFFFFFFF00- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
3879 0001738D 0000000000000FF     <1>
3880 00017394 FFFFFFFFFFFFF0F0F0- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
3880 0001739D F0F0F0F0F0F0F0      <1>
3881 000173A4 F0F0F0F00F0F0F0F0F- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
3881 000173AD 0F0F0F0F0F0F0F      <1>
3882 000173B4 0F0FFFFFFFFFFFFFFF- <1>     db  00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3882 000173BD 00000000000000      <1>
3883 000173C4 0000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h, 000h
3883 000173CD DC760000000000      <1>
3884 000173D4 00007CC6FCC6C6FCC0- <1>     db  000h, 000h, 07ch, 0c6h, 0fch, 0c6h, 0c6h, 0fch, 0c0h, 0c0h, 040h, 000h, 000h, 000h, 0feh, 0c6h
3884 000173DD C040000000FEC6      <1>
```

```
3885 000173E4 C6C0C0C0C0C0C00000- <1>     db  0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 06ch
3885 000173ED 0000000000FE6C      <1>
3886 000173F4 6C6C6C6C6C00000000- <1>     db  06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h
3886 000173FD 00FEC660301830      <1>
3887 00017404 60C6FE000000000000- <1>     db  060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h
3887 0001740D 00007ED8D8D8D8      <1>
3888 00017414 700000000000000066- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h
3888 0001741D 6666667C6060C0      <1>
3889 00017424 00000000000076DC18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h
3889 0001742D 18181818000000      <1>
3890 00017434 00007E183C6666663C- <1>     db  000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h, 000h
3890 0001743D 187E0000000000      <1>
3891 00017444 386CC6C6FEC6C66C38- <1>     db  038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch
3891 0001744D 0000000000386C      <1>
3892 00017454 C6C6C66C6C6CEE0000- <1>     db  0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h, 000h, 01eh, 030h, 018h, 00ch
3892 0001745D 0000001E30180C      <1>
3893 00017464 3E6666663C00000000- <1>     db  03eh, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh
3893 0001746D 000000007EDBDB      <1>
3894 00017474 7E00000000000000003- <1>     db  07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h
3894 0001747D 067EDBDBF37E60      <1>
3895 00017484 C000000000001C3060- <1>     db  0c0h, 000h, 000h, 000h, 000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 030h, 01ch, 000h
3895 0001748D 607C6060301C00      <1>
3896 00017494 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h
3896 0001749D C6C6C6C6000000      <1>
3897 000174A4 000000FE0000FE0000- <1>     db  000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
3897 000174AD FE000000000000      <1>
3898 000174B4 0018187E18180000FF- <1>     db  000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 030h, 018h
3898 000174BD 00000000003018      <1>
3899 000174C4 0C060C1830007E0000- <1>     db  00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h, 060h
3899 000174CD 0000000C183060      <1>
3900 000174D4 30180C007E00000000- <1>     db  030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h
3900 000174DD 000E1B1B181818      <1>
3901 000174E4 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h
3901 000174ED 1818181818D8D8      <1>
3902 000174F4 700000000000001818- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h
3902 000174FD 007E0018180000      <1>
3903 00017504 00000000000076DC00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h
3903 0001750D 76DC0000000000      <1>
3904 00017514 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3904 0001751D 00000000000000      <1>
3905 00017524 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3905 0001752D 00000000000000      <1>
3906 00017534 000001800000000000- <1>     db  000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 00ch
3906 0001753D 00000F0C0C0C0C      <1>
3907 00017544 0CEC6C3C1C00000000- <1>     db  00ch, 0ech, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
3907 0001754D D86C6C6C6C6C00      <1>
3908 00017554 0000000000000070D8- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h
3908 0001755D 3060C8F8000000      <1>
3909 00017564 000000000000000007C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h
3909 0001756D 7C7C7C7C7C0000      <1>
3910 00017574 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3910 0001757D 00000000000000      <1>
3911                              <1> vgafont16:
3912 00017584 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3912 0001758D 00000000000000      <1>
3913 00017594 00007E81A58181BD99- <1>     db  000h, 000h, 07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 081h, 07eh, 000h, 000h, 000h, 000h
3913 0001759D 81817E00000000      <1>
3914 000175A4 00007EFFDBFFFFC3E7- <1>     db  000h, 000h, 07eh, 0ffh, 0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 0ffh, 07eh, 000h, 000h, 000h, 000h
3914 000175AD FFFF7E00000000      <1>
3915 000175B4 000000006CFEFEFEFE- <1>     db  000h, 000h, 000h, 000h, 06ch, 0feh, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h
3915 000175BD 7C381000000000      <1>
3916 000175C4 0000000010387CFE7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h
3916 000175CD 38100000000000      <1>
3917 000175D4 000000183C3CE7E7E7- <1>     db  000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
3917 000175DD 18183C00000000      <1>
3918 000175E4 000000183C7EFFFF7E- <1>     db  000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
3918 000175ED 18183C00000000      <1>
3919 000175F4 000000000000183C3C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
3919 000175FD 18000000000000      <1>
3920 00017604 FFFFFFFFFFFFE7C3C3- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3920 0001760D E7FFFFFFFFFFFF      <1>
3921 00017614 00000000003C664242- <1>     db  000h, 000h, 000h, 000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h
3921 0001761D 663C0000000000      <1>
3922 00017624 FFFFFFFFFFC399BDBD- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3922 0001762D 99C3FFFFFFFFFF      <1>
3923 00017634 00001E0E1A3278CCCC- <1>     db  000h, 000h, 01eh, 00eh, 01ah, 032h, 078h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
3923 0001763D CCCC7800000000      <1>
3924 00017644 00003C666666663C18- <1>     db  000h, 000h, 03ch, 066h, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
3924 0001764D 7E181800000000      <1>
3925 00017654 00003F333F30303030- <1>     db  000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 030h, 070h, 0f0h, 0e0h, 000h, 000h, 000h, 000h
3925 0001765D 70F0E000000000      <1>
3926 00017664 00007F637F63636363- <1>     db  000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h, 000h, 000h, 000h
3926 0001766D 67E7E6C0000000      <1>
3927 00017674 0000001818DB3CE73C- <1>     db  000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h, 000h
3927 0001767D DB1818000000000     <1>
3928 00017684 0080C0E0F0F8FEF8F0- <1>     db  000h, 080h, 0c0h, 0e0h, 0f0h, 0f8h, 0feh, 0f8h, 0f0h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h
3928 0001768D E0C08000000000      <1>
3929 00017694 0002060E1E3EFE3E1E- <1>     db  000h, 002h, 006h, 00eh, 01eh, 03eh, 0feh, 03eh, 01eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
3929 0001769D 0E060200000000      <1>
3930 000176A4 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
3930 000176AD 3C180000000000      <1>
3931 000176B4 0006666666666666- <1>     db  000h, 000h, 066h, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h
3931 000176BD 00666600000000      <1>
3932 000176C4 00007FDBDBDB7B1B1B- <1>     db  000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 01bh, 01bh, 000h, 000h, 000h, 000h
3932 000176CD 1B1B1B00000000      <1>
3933 000176D4 007CC660386CC6C66C- <1>     db  000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h
3933 000176DD 380CC67C000000      <1>
3934 000176E4 00000000000000000FE- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 0feh, 000h, 000h, 000h, 000h
3934 000176ED FEFEFE00000000      <1>
3935 000176F4 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
3935 000176FD 3C187E00000000      <1>
3936 00017704 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
3936 0001770D 18181800000000      <1>
3937 00017714 000018181818181818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h
3937 0001771D 7E3C1800000000      <1>
3938 00017724 0000000000180CFE0C- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
3938 0001772D 18000000000000      <1>
3939 00017734 00000000003060FE60- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h
3939 0001773D 30000000000000      <1>
3940 00017744 000000000000C0C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
3940 0001774D FE000000000000      <1>
```

```
3941 00017754 00000000002466FF66- <1>     db  000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h
3941 0001775D 24000000000000      <1>
3942 00017764 0000000001038387C7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h, 000h, 000h, 000h
3942 0001776D FEFE0000000000      <1>
3943 00017774 00000000FEFE7C7C38- <1>     db  000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h, 000h
3943 0001777D 38100000000000      <1>
3944 00017784 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3944 0001778D 00000000000000      <1>
3945 00017794 0000183C3C3C181818- <1>     db  000h, 000h, 018h, 03ch, 03ch, 03ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
3945 0001779D 00181800000000      <1>
3946 000177A4 006666662400000000- <1>     db  000h, 066h, 066h, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3946 000177AD 00000000000000      <1>
3947 000177B4 0000006C6CFE6C6C6C- <1>     db  000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 000h
3947 000177BD FE6C6C00000000      <1>
3948 000177C4 18187CC6C2C07C0606- <1>     db  018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h, 006h, 086h, 0c6h, 07ch, 018h, 018h, 000h, 000h
3948 000177CD 86C67C18180000      <1>
3949 000177D4 00000000C2C60C1830- <1>     db  000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 060h, 0c6h, 086h, 000h, 000h, 000h, 000h
3949 000177DD 60C68600000000      <1>
3950 000177E4 0000386C6C3876DCCC- <1>     db  000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
3950 000177ED CCCC7600000000      <1>
3951 000177F4 0030303060000000000- <1>     db  000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3951 000177FD 00000000000000      <1>
3952 00017804 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h
3952 0001780D 30180C00000000      <1>
3953 00017814 000030180C0C0C0C0C- <1>     db  000h, 000h, 030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h
3953 0001781D 0C183000000000      <1>
3954 00017824 0000000000663CFF3C- <1>     db  000h, 000h, 000h, 000h, 000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h
3954 0001782D 66000000000000      <1>
3955 00017834 0000000000018187E18- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h
3955 0001783D 18000000000000      <1>
3956 00017844 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h
3956 0001784D 18181830000000      <1>
3957 00017854 000000000000000FE00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3957 0001785D 00000000000000      <1>
3958 00017864 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
3958 0001786D 00181800000000      <1>
3959 00017874 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
3959 0001787D 60C08000000000      <1>
3960 00017884 00003C66C3C3DBDBC3- <1>     db  000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h, 000h, 000h
3960 0001788D C3663C00000000      <1>
3961 00017894 0000183878181818- <1>     db  000h, 000h, 018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h
3961 0001789D 18187E00000000      <1>
3962 000178A4 00007CC6060C183060- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 0c6h, 0feh, 000h, 000h, 000h, 000h
3962 000178AD C0C6FE00000000      <1>
3963 000178B4 00007CC606063C0606- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 006h, 03ch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3963 000178BD 06C67C00000000      <1>
3964 000178C4 00000C1C3C6CCCFE0C- <1>     db  000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h
3964 000178CD 0C0C1E00000000      <1>
3965 000178D4 0000FEC0C0C0FC0606- <1>     db  000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3965 000178DD 06C67C00000000      <1>
3966 000178E4 00003860C0C0FCC6C6- <1>     db  000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3966 000178ED C6C67C00000000      <1>
3967 000178F4 0000FEC606060C1830- <1>     db  000h, 000h, 0feh, 0c6h, 006h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h, 000h
3967 000178FD 30303000000000      <1>
3968 00017904 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3968 0001790D C6C67C00000000      <1>
3969 00017914 00007CC6C6C67E0606- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h
3969 0001791D 060C7800000000      <1>
3970 00017924 0000000018180000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
3970 0001792D 18180000000000      <1>
3971 00017934 0000000018180000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h
3971 0001793D 18183000000000      <1>
3972 00017944 000000060C18306030- <1>     db  000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 006h, 000h, 000h, 000h, 000h
3972 0001794D 180C0600000000      <1>
3973 00017954 00000000007E00007E- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3973 0001795D 00000000000000      <1>
3974 00017964 0000006030180C060C- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h, 000h, 000h, 000h
3974 0001796D 18306000000000      <1>
3975 00017974 00007CC6C60C181818- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
3975 0001797D 00181800000000      <1>
3976 00017984 0000007CC6C6DEDEDE- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h
3976 0001798D DCC07C00000000      <1>
3977 00017994 000010386CC6C6FEC6- <1>     db  000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
3977 0001799D C6C6C600000000      <1>
3978 000179A4 0000FC6666667C6666- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 066h, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h
3978 000179AD 6666FC00000000      <1>
3979 000179B4 00003C66C2C0C0C0C0- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h
3979 000179BD C2663C00000000      <1>
3980 000179C4 0000F86C6666666666- <1>     db  000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h, 066h, 066h, 06ch, 0f8h, 000h, 000h, 000h, 000h
3980 000179CD 666CF800000000      <1>
3981 000179D4 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
3981 000179DD 6266FE00000000      <1>
3982 000179E4 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
3982 000179ED 6060F000000000      <1>
3983 000179F4 00003C66C2C0C0DEC6- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 0c6h, 066h, 03ah, 000h, 000h, 000h, 000h
3983 000179FD C6663A00000000      <1>
3984 00017A04 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
3984 00017A0D C6C6C600000000      <1>
3985 00017A14 00003C181818181818- <1>     db  000h, 000h, 03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
3985 00017A1D 18183C00000000      <1>
3986 00017A24 00001E0C0C0C0C0CCC- <1>     db  000h, 000h, 01eh, 00ch, 00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
3986 00017A2D CCCC7800000000      <1>
3987 00017A34 0000E666666C78786C- <1>     db  000h, 000h, 0e6h, 066h, 066h, 06ch, 078h, 078h, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
3987 00017A3D 6666E600000000      <1>
3988 00017A44 0000F06060606060C0- <1>     db  000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
3988 00017A4D 6266FE00000000      <1>
3989 00017A54 0000C3E7FFFFDBC3C3- <1>     db  000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
3989 00017A5D C3C3C300000000      <1>
3990 00017A64 0000C6E6F6FEDECEC6- <1>     db  000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
3990 00017A6D C6C6C600000000      <1>
3991 00017A74 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3991 00017A7D C6C67C00000000      <1>
3992 00017A84 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
3992 00017A8D 6060F000000000      <1>
3993 00017A94 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h
3993 00017A9D D6DE7C0C0E0000      <1>
3994 00017AA4 0000FC6666667C6C66- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 06ch, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
3994 00017AAD 6666E600000000      <1>
3995 00017AB4 00007CC6C660380C06- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 060h, 038h, 00ch, 006h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3995 00017ABD C6C67C00000000      <1>
3996 00017AC4 0000FFDB9918181818- <1>     db  000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
```

```
3996 00017ACD 18183C00000000      <1>
3997 00017AD4 0000C6C6C6C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3997 00017ADD C6C67C00000000      <1>
3998 00017AE4 0000C3C3C3C3C3C3C3- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
3998 00017AED 663C1800000000      <1>
3999 00017AF4 0000C3C3C3C3C3DBDB- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 000h
3999 00017AFD FF666600000000      <1>
4000 00017B04 0000C3C3663C18183C- <1>     db  000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
4000 00017B0D 66C3C300000000      <1>
4001 00017B14 0000C3C3C3663C1818- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4001 00017B1D 18183C00000000      <1>
4002 00017B24 0000FFC3860C183060- <1>     db  000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h
4002 00017B2D C1C3FF00000000      <1>
4003 00017B34 00003C303030303030- <1>     db  000h, 000h, 03ch, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h
4003 00017B3D 30303C00000000      <1>
4004 00017B44 00000080C0E070381C- <1>     db  000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
4004 00017B4D 0E060200000000      <1>
4005 00017B54 00003C0C0C0C0C0C0C- <1>     db  000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 03ch, 000h, 000h, 000h, 000h
4005 00017B5D 0C0C3C00000000      <1>
4006 00017B64 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4006 00017B6D 00000000000000      <1>
4007 00017B74 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h
4007 00017B7D 00000000FF0000      <1>
4008 00017B84 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4008 00017B8D 00000000000000      <1>
4009 00017B94 0000000000780C7CCC- <1>     db  000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4009 00017B9D CCCC7600000000      <1>
4010 00017BA4 0000E06060786C6666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 078h, 06ch, 066h, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h
4010 00017BAD 66667C00000000      <1>
4011 00017BB4 00000000007CC6C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c0h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4011 00017BBD C0C67C00000000      <1>
4012 00017BC4 00001C0C0C3C6CCCCC- <1>     db  000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4012 00017BCD CCCC7600000000      <1>
4013 00017BD4 00000000007CC6FEC0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4013 00017BDD C0C67C00000000      <1>
4014 00017BE4 0000386C6460F06060- <1>     db  000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
4014 00017BED 6060F000000000      <1>
4015 00017BF4 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
4015 00017BFD CCCC7C0CCC7800      <1>
4016 00017C04 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
4016 00017C0D 6666E600000000      <1>
4017 00017C14 000018180038181818- <1>     db  000h, 000h, 018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4017 00017C1D 18183C00000000      <1>
4018 00017C24 00000606000E060606- <1>     db  000h, 000h, 006h, 006h, 000h, 00eh, 006h, 006h, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h
4018 00017C2D 06060666663C00      <1>
4019 00017C34 0000E06060666C7878- <1>     db  000h, 000h, 0e0h, 060h, 060h, 066h, 06ch, 078h, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h
4019 00017C3D 6C66E600000000      <1>
4020 00017C44 000038181818181818- <1>     db  000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4020 00017C4D 18183C00000000      <1>
4021 00017C54 0000000000E6FFDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h
4021 00017C5D DBDBDB00000000      <1>
4022 00017C64 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
4022 00017C6D 66666600000000      <1>
4023 00017C74 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4023 00017C7D C6C67C00000000      <1>
4024 00017C84 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h
4024 00017C8D 66667C6060F000      <1>
4025 00017C94 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h
4025 00017C9D CCCC7C0C0C1E00      <1>
4026 00017CA4 0000000000DC766660- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 076h, 066h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
4026 00017CAD 6060F000000000      <1>
4027 00017CB4 00000000007CC66038- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h
4027 00017CBD 0CC67C00000000      <1>
4028 00017CC4 0000103030FC303030- <1>     db  000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h, 030h, 030h, 036h, 01ch, 000h, 000h, 000h, 000h
4028 00017CCD 30361C00000000      <1>
4029 00017CD4 0000000000CCCCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4029 00017CDD CCCC7600000000      <1>
4030 00017CE4 0000000000C3C3C3C3- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
4030 00017CED 663C1800000000      <1>
4031 00017CF4 0000000000C3C3C3DB- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h
4031 00017CFD DBFF6600000000      <1>
4032 00017D04 0000000000C3663C18- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h
4032 00017D0D 3C66C300000000      <1>
4033 00017D14 0000000000C6C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h
4033 00017D1D C6C67E060CF800      <1>
4034 00017D24 0000000000FECC1830- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0cch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
4034 00017D2D 60C6FE00000000      <1>
4035 00017D34 00000E181818701818- <1>     db  000h, 000h, 00eh, 018h, 018h, 018h, 070h, 018h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h
4035 00017D3D 18180E00000000      <1>
4036 00017D44 0000181818180018 18- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4036 00017D4D 18181800000000      <1>
4037 00017D54 0000701818180E1818- <1>     db  000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h, 018h, 070h, 000h, 000h, 000h, 000h
4037 00017D5D 18187000000000      <1>
4038 00017D64 000076DC0000000000- <1>     db  000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4038 00017D6D 00000000000000      <1>
4039 00017D74 0000000010386CC6C6- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h
4039 00017D7D C6FE0000000000      <1>
4040 00017D84 00003C66C2C0C0C0C2- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h
4040 00017D8D 663C0C067C0000      <1>
4041 00017D94 0000CC0000CCCCCCCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4041 00017D9D CCCC7600000000      <1>
4042 00017DA4 000C1830007CC6FEC0- <1>     db  000h, 00ch, 018h, 030h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4042 00017DAD C0C67C00000000      <1>
4043 00017DB4 0010386C00780C7CCC- <1>     db  000h, 010h, 038h, 06ch, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4043 00017DBD CCCC7600000000      <1>
4044 00017DC4 0000CC0000780C7CCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4044 00017DCD CCCC7600000000      <1>
4045 00017DD4 0060301800780C7CCC- <1>     db  000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4045 00017DDD CCCC7600000000      <1>
4046 00017DE4 00386C3800780C7CCC- <1>     db  000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4046 00017DED CCCC7600000000      <1>
4047 00017DF4 000000003C66606066- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 060h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h, 000h
4047 00017DFD 3C0C063C000000      <1>
4048 00017E04 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4048 00017E0D C0C67C00000000      <1>
4049 00017E14 0000C600007CC6FEC0- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4049 00017E1D C0C67C00000000      <1>
4050 00017E24 00603018007CC6FEC0- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4050 00017E2D C0C67C00000000      <1>
4051 00017E34 000066000038181818- <1>     db  000h, 000h, 066h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4051 00017E3D 18183C00000000      <1>
```

```
4052 00017E44 00183C660038181818- <1>     db  000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4052 00017E4D 18183C00000000      <1>
4053 00017E54 006030180038181818- <1>     db  000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4053 00017E5D 18183C00000000      <1>
4054 00017E64 00C60010386CC6C6FE- <1>     db  000h, 0c6h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4054 00017E6D C6C6C600000000      <1>
4055 00017E74 386C3800386CC6C6FE- <1>     db  038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4055 00017E7D C6C6C600000000      <1>
4056 00017E84 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 060h, 066h, 0feh, 000h, 000h, 000h, 000h
4056 00017E8D 6066FE00000000      <1>
4057 00017E94 00000000006E3B1B7E- <1>     db  000h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h
4057 00017E9D D8DC7700000000      <1>
4058 00017EA4 00003E6CCCCCFECCCC- <1>     db  000h, 000h, 03eh, 06ch, 0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h
4058 00017EAD CCCCCE00000000      <1>
4059 00017EB4 0010386C007CC6C6C6- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4059 00017EBD C6C67C00000000      <1>
4060 00017EC4 0000C600007CC6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4060 00017ECD C6C67C00000000      <1>
4061 00017ED4 00603018007CC6C6C6- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4061 00017EDD C6C67C00000000      <1>
4062 00017EE4 003078CC00CCCCCCCC- <1>     db  000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4062 00017EED CCCC7600000000      <1>
4063 00017EF4 0060301800CCCCCCCC- <1>     db  000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4063 00017EFD CCCC7600000000      <1>
4064 00017F04 0000C60000C6C6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h
4064 00017F0D C6C67E060C7800      <1>
4065 00017F14 00C6007CC6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4065 00017F1D C6C67C00000000      <1>
4066 00017F24 00C600C6C6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4066 00017F2D C6C67C00000000      <1>
4067 00017F34 0018187EC3C0C0C0C3- <1>     db  000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
4067 00017F3D 7E181800000000      <1>
4068 00017F44 00386C6460F0606060- <1>     db  000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0e6h, 0fch, 000h, 000h, 000h, 000h
4068 00017F4D 60E6FC00000000      <1>
4069 00017F54 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4069 00017F5D 18181800000000      <1>
4070 00017F64 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h, 000h
4070 00017F6D 6666F300000000      <1>
4071 00017F74 000E1B1818187E1818- <1>     db  000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h, 000h
4071 00017F7D 181818D8700000      <1>
4072 00017F84 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4072 00017F8D CCCC7600000000      <1>
4073 00017F94 000C18300038181818- <1>     db  000h, 00ch, 018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4073 00017F9D 18183C00000000      <1>
4074 00017FA4 00183060007CC6C6C6- <1>     db  000h, 018h, 030h, 060h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4074 00017FAD C6C67C00000000      <1>
4075 00017FB4 0018306000CCCCCCCC- <1>     db  000h, 018h, 030h, 060h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4075 00017FBD CCCC7600000000      <1>
4076 00017FC4 000076DC00DC666666- <1>     db  000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
4076 00017FCD 66666600000000      <1>
4077 00017FD4 76DC00C6E6F6FEDECE- <1>     db  076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4077 00017FDD C6C6C600000000      <1>
4078 00017FE4 003C6C6C3E007E0000- <1>     db  000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4078 00017FED 00000000000000      <1>
4079 00017FF4 00386C6C38007C0000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4079 00017FFD 00000000000000      <1>
4080 00018004 0000303000303060C0- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c0h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4080 0001800D C6C67C00000000      <1>
4081 00018014 000000000000FEC0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h
4081 0001801D C0C00000000000      <1>
4082 00018024 000000000000FE0606- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 006h, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h
4082 0001802D 06060000000000      <1>
4083 00018034 00C0C0C2C6CC183060- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh, 000h, 000h
4083 0001803D CE9B060C1F0000      <1>
4084 00018044 00C0C0C2C6CC183066- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h, 006h, 000h, 000h
4084 0001804D CE963E06060000      <1>
4085 00018054 000018180018183C- <1>     db  000h, 000h, 018h, 018h, 000h, 018h, 018h, 018h, 03ch, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h
4085 0001805D 3C3C1800000000      <1>
4086 00018064 0000000000366CD86C- <1>     db  000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h, 000h, 000h, 000h
4086 0001806D 36000000000000      <1>
4087 00018074 0000000000D86C366C- <1>     db  000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h, 000h
4087 0001807D D8000000000000      <1>
4088 00018084 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h
4088 0001808D 44114411441144      <1>
4089 00018094 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
4089 0001809D AA55AA55AA55AA      <1>
4090 000180A4 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h
4090 000180AD 77DD77DD77DD77      <1>
4091 000180B4 181818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4091 000180BD 18181818181818      <1>
4092 000180C4 1818181818181818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4092 000180CD 18181818181818      <1>
4093 000180D4 1818181818F818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4093 000180DD 18181818181818      <1>
4094 000180E4 363636363636F636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4094 000180ED 36363636363636      <1>
4095 000180F4 000000000000FE36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4095 000180FD 36363636363636      <1>
4096 00018104 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4096 0001810D 18181818181818      <1>
4097 00018114 3636363636F606F636- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4097 0001811D 36363636363636      <1>
4098 00018124 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4098 0001812D 36363636363636      <1>
4099 00018134 0000000000FE06F636- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4099 0001813D 36363636363636      <1>
4100 00018144 3636363636F606FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4100 0001814D 00000000000000      <1>
4101 00018154 36363636363636FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4101 0001815D 00000000000000      <1>
4102 00018164 1818181818F818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4102 0001816D 00000000000000      <1>
4103 00018174 000000000000F818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4103 0001817D 18181818181818      <1>
4104 00018184 18181818181818F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4104 0001818D 00000000000000      <1>
4105 00018194 18181818181818FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4105 0001819D 00000000000000      <1>
4106 000181A4 000000000000FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4106 000181AD 18181818181818      <1>
4107 000181B4 18181818181818F18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
```

```
4107 000181BD 18181818181818      <1>
4108 000181C4 000000000000000FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4108 000181CD 00000000000000      <1>
4109 000181D4 181818181818FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4109 000181DD 18181818181818      <1>
4110 000181E4 18181818181F181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4110 000181ED 18181818181818      <1>
4111 000181F4 363636363636363736- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4111 000181FD 36363636363636      <1>
4112 00018204 363636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4112 0001820D 00000000000000      <1>
4113 00018214 00000000003F303736- <1>     db  000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4113 0001821D 36363636363636      <1>
4114 00018224 3636363636F700FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4114 0001822D 00000000000000      <1>
4115 00018234 0000000000FF00F736- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4115 0001823D 36363636363636      <1>
4116 00018244 363636363637303736- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4116 0001824D 36363636363636      <1>
4117 00018254 0000000000FF00FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4117 0001825D 00000000000000      <1>
4118 00018264 3636363636F700F736- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4118 0001826D 36363636363636      <1>
4119 00018274 1818181818FF00FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4119 0001827D 00000000000000      <1>
4120 00018284 363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4120 0001828D 00000000000000      <1>
4121 00018294 0000000000FF00FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4121 0001829D 18181818181818      <1>
4122 000182A4 000000000000000FF36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4122 000182AD 36363636363636      <1>
4123 000182B4 363636363636363F00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4123 000182BD 00000000000000      <1>
4124 000182C4 18181818181F181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4124 000182CD 00000000000000      <1>
4125 000182D4 000000000001F181F18- <1>     db  000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4125 000182DD 18181818181818      <1>
4126 000182E4 000000000000003F36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4126 000182ED 36363636363636      <1>
4127 000182F4 363636363636FF36- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4127 000182FD 36363636363636      <1>
4128 00018304 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4128 0001830D 18181818181818      <1>
4129 00018314 18181818181818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4129 0001831D 00000000000000      <1>
4130 00018324 000000000000001F18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4130 0001832D 18181818181818      <1>
4131 00018334 FFFFFFFFFFFFFFFFFF- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
4131 0001833D FFFFFFFFFFFFFF      <1>
4132 00018344 000000000000000FFFF- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
4132 0001834D FFFFFFFFFFFFFF      <1>
4133 00018354 F0F0F0F0F0F0F0F0F0- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
4133 0001835D F0F0F0F0F0F0F0      <1>
4134 00018364 0F0F0F0F0F0F0F0F0F- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
4134 0001836D 0F0F0F0F0F0F0F      <1>
4135 00018374 FFFFFFFFFFFFFF0000- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4135 0001837D 00000000000000      <1>
4136 00018384 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h
4136 0001838D D8DC7600000000      <1>
4137 00018394 000078CCCCCCD8CCC6- <1>     db  000h, 000h, 078h, 0cch, 0cch, 0cch, 0d8h, 0cch, 0c6h, 0c6h, 0c6h, 0cch, 000h, 000h, 000h, 000h
4137 0001839D C6C6CC00000000      <1>
4138 000183A4 0000FEC6C6C0C0C0C0- <1>     db  000h, 000h, 0feh, 0c6h, 0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h
4138 000183AD C0C0C000000000      <1>
4139 000183B4 00000000FE6C6C6C6C- <1>     db  000h, 000h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h
4139 000183BD 6C6C6C00000000      <1>
4140 000183C4 000000FEC660301830- <1>     db  000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
4140 000183CD 60C6FE00000000      <1>
4141 000183D4 00000000007ED8D8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
4141 000183DD D8D870000000000     <1>
4142 000183E4 000000066666666666- <1>     db  000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h, 000h, 000h, 000h
4142 000183ED 7C6060C0000000      <1>
4143 000183F4 0000000076DC181818- <1>     db  000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4143 000183FD 18181800000000      <1>
4144 00018404 0000007E183C666666- <1>     db  000h, 000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
4144 0001840D 3C187E00000000      <1>
4145 00018414 000000386CC6C6FEC6- <1>     db  000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h
4145 0001841D C66C3800000000      <1>
4146 00018424 0000386CC6C6C66C6C- <1>     db  000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h
4146 0001842D 6C6CEE00000000      <1>
4147 00018434 00001E30180C3E6666- <1>     db  000h, 000h, 01eh, 030h, 018h, 00ch, 03eh, 066h, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h
4147 0001843D 66663C00000000      <1>
4148 00018444 00000000007EDBDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh, 0dbh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h
4148 0001844D 7E000000000000      <1>
4149 00018454 00000003067EDBDBF3- <1>     db  000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h, 0c0h, 000h, 000h, 000h, 000h
4149 0001845D 7E60C000000000      <1>
4150 00018464 00001C3060607C6060- <1>     db  000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 060h, 030h, 01ch, 000h, 000h, 000h, 000h
4150 0001846D 60301C00000000      <1>
4151 00018474 0000007CC6C6C6C6C6- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4151 0001847D C6C6C600000000      <1>
4152 00018484 00000000FE0000FE00- <1>     db  000h, 000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h
4152 0001848D 00FE0000000000      <1>
4153 00018494 0000000018187E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h
4153 0001849D 0000FF00000000      <1>
4154 000184A4 00000030180C060C18- <1>     db  000h, 000h, 000h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h
4154 000184AD 30007E00000000      <1>
4155 000184B4 0000000C1830603018- <1>     db  000h, 000h, 000h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h
4155 000184BD 0C007E00000000      <1>
4156 000184C4 00000E1B1B18181818- <1>     db  000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4156 000184CD 18181818181818      <1>
4157 000184D4 1818181818181818D8- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
4157 000184DD D8D870000000000     <1>
4158 000184E4 000000001818007E00- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
4158 000184ED 18180000000000      <1>
4159 000184F4 000000000076DC0076- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h
4159 000184FD DC0000000000000     <1>
4160 00018504 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4160 0001850D 00000000000000      <1>
4161 00018514 000000000000001818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4161 0001851D 00000000000000      <1>
4162 00018524 000000000000000018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4162 0001852D 00000000000000      <1>
```

```
4163 00018534 000F0C0C0C0C0CEC6C- <1>     db  000h, 00fh, 00ch, 00ch, 00ch, 00ch, 00ch, 0ech, 06ch, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h
4163 0001853D 6C3C1C00000000      <1>
4164 00018544 00D86C6C6C6C6C0000- <1>     db  000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4164 0001854D 00000000000000      <1>
4165 00018554 0070D83060C8F80000- <1>     db  000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4165 0001855D 00000000000000      <1>
4166 00018564 000000007C7C7C7C7C- <1>     db  000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h, 000h, 000h, 000h
4166 0001856D 7C7C0000000000      <1>
4167 00018574 0000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4167 0001857D 00000000000000      <1>
4168                              <1>
4169                              <1> ; 01/01/2021 (TRDOS 386 v2.0.3)
4170                              <1>
4171                              <1> %if 0
4172                              <1>
4173                              <1> vgafont14alt:
4174                              <1>     db  01dh, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 022h
4175                              <1>     db  000h, 063h, 063h, 063h, 022h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02bh, 000h
4176                              <1>     db  000h, 000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 02dh, 000h, 000h
4177                              <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 04dh, 000h, 000h, 0c3h
4178                              <1>     db  0e7h, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh
4179                              <1>     db  099h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h
4180                              <1>     db  0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h
4181                              <1>     db  0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h
4182                              <1>     db  03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
4183                              <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 061h
4184                              <1>     db  0c3h, 0ffh, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh
4185                              <1>     db  0dbh, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
4186                              <1>     db  000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h
4187                              <1>     db  000h, 000h, 091h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h
4188                              <1>     db  000h, 09bh, 000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h
4189                              <1>     db  09dh, 000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 000h, 000h, 000h, 09eh
4190                              <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 0f3h, 000h, 000h, 000h, 0f1h, 000h
4191                              <1>     db  000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 0ffh, 000h, 000h, 000h, 0f6h, 000h, 000h
4192                              <1>     db  018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
4193                              <1> vgafont16alt:
4194                              <1>     db  01dh, 000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h
4195                              <1>     db  000h, 030h, 000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h
4196                              <1>     db  000h, 000h, 04dh, 000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h
4197                              <1>     db  000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch
4198                              <1>     db  000h, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch
4199                              <1>     db  018h, 000h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh
4200                              <1>     db  066h, 066h, 000h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch
4201                              <1>     db  066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
4202                              <1>     db  018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h
4203                              <1>     db  030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h
4204                              <1>     db  0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h
4205                              <1>     db  0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h
4206                              <1>     db  000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h, 078h, 000h, 000h, 000h
4207                              <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h, 091h, 000h, 000h
4208                              <1>     db  000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h, 09bh, 000h
4209                              <1>     db  018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 09dh
4210                              <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4211                              <1>     db  09eh, 000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h
4212                              <1>     db  000h, 0abh, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh
4213                              <1>     db  000h, 000h, 0ach, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h
4214                              <1>     db  006h, 000h, 000h, 000h
4215                              <1>
4216                              <1> %endif
3434
3435                                    ; 20/11/2020
3436                                    vbe2_bochs_vbios:
3437                                    ;     db "BOCHS/QEMU"
3438 00018584 424F4348532F51454D-          db "BOCHS/QEMU/VIRTUALBOX"  ; 26/11/2020
3438 0001858D 552F5649525455414C-
3438 00018596 424F58
3439                                    ;vbe_vnumber equ vbe2_bochs_vbios + 28 ; "3" or "2"
3440                                    ; 26/11/2020
3441                                    vbe_vnumber equ vbe2_bochs_vbios + 30 ; "3" or "2"
3442                                    ; 15/11/2020
3443                                    vesa_vbe3_bios_msg:
3444                                          ;db " VESA VBE version 3 Video BIOS ..."
3445 00018599 205645534120564245-          db " VESA VBE3 Video BIOS ..." ; 26/11/2020
3445 000185A2 3320566964656F2042-
3445 000185AB 494F53202E2E2E
3446 000185B2 0D0A0D0A00                  db 0Dh, 0Ah, 0Dh, 0Ah, 0
3447
3448                                    ; 28/02/2021
3449 000185B7 18                        truecolor: db 24
3450
3451                                    align 2
3452
3453                                    ; EPOCH Variables
3454                                    ; 13/04/2015 - Retro UNIX 386 v1 Beginning
3455                                    ; 09/04/2013 epoch variables
3456                                    ; Retro UNIX 8086 v1 Prototype: UNIXCOPY.ASM, 10/03/2013
3457                                    ;
3458 000185B8 B207                     year:       dw 1970
3459 000185BA 0100                     month:      dw 1
3460 000185BC 0100                     day:  dw 1
3461 000185BE 0000                     hour:       dw 0
3462 000185C0 0000                     minute: dw 0
3463 000185C2 0000                     second: dw 0
3464
3465                                    DMonth:
3466 000185C4 0000                            dw 0
3467 000185C6 1F00                            dw 31
3468 000185C8 3B00                            dw 59
3469 000185CA 5A00                            dw 90
3470 000185CC 7800                            dw 120
3471 000185CE 9700                            dw 151
3472 000185D0 B500                            dw 181
3473 000185D2 D400                            dw 212
3474 000185D4 F300                            dw 243
3475 000185D6 1101                            dw 273
3476 000185D8 3001                            dw 304
3477 000185DA 4E01                            dw 334
3478
```

```
                                    ; 15/11/2020
000185DC 00                         db    0
                                    kernel_version_msg: ; 20/11/2020
000185DD 5452444F5320283338-        db    "TRDOS (386) Kernel v2.0.3 by Erdogan Tan"
000185E6 3629204B65726E656C-
000185EF 2076322E302E332062-
000185F8 79204572646F67616E-
00018601 2054616E
00018605 00                         db    0

                                    ; 20/02/2017
                                    KERNELFSIZE  equ $  ; 04/07/2016

                                    bss_start:

                                    ABSOLUTE bss_start

00018606 ????                       alignb 8 ; 25/12/2016

                                          ; 15/04/2016
                                          ; TRDOS 386 (TRDOS v2.0)
                                          ;      80 interrupts
                                          ; 11/03/2015
                                          ; Interrupt Descriptor Table (20/08/2014)
                                    idt:
                                          ;resb  64*8 ; INT 0 to INT 3Fh
                                          ; 15/04/2016
00018608 <res 280h>                       resb   80*8 ; INT 0 to INT 4Fh

                                    idt_end:

                                    ;alignb 4

                                    task_state_segment:
                                          ; 24/03/2015
00018888 ????                       tss.link:   resw 1
0001888A ????                                   resw 1
                                    ; tss offset 4
0001888C ????????                   tss.esp0:   resd 1
00018890 ????                       tss.ss0:    resw 1
00018892 ????                                   resw 1
00018894 ????????                   tss.esp1:   resd 1
00018898 ????                       tss.ss1:    resw 1
0001889A ????                                   resw 1
0001889C ????????                   tss.esp2:   resd 1
000188A0 ????                       tss.ss2:    resw 1
000188A2 ????                                   resw 1
                                    ; tss offset 28
000188A4 ????????                   tss.CR3:    resd 1
000188A8 ????????                   tss.eip:    resd 1
000188AC ????????                   tss.eflags: resd 1
                                    ; tss offset 40
000188B0 ????????                   tss.eax:    resd 1
000188B4 ????????                   tss.ecx:    resd 1
000188B8 ????????                   tss.edx:    resd 1
000188BC ????????                   tss.ebx:    resd 1
000188C0 ????????                   tss.esp:    resd 1
000188C4 ????????                   tss.ebp:    resd 1
000188C8 ????????                   tss.esi:    resd 1
000188CC ????????                   tss.edi:    resd 1
                                    ; tss offset 72
000188D0 ????                       tss.ES:     resw 1
000188D2 ????                                   resw 1
000188D4 ????                       tss.CS:         resw 1
000188D6 ????                                   resw 1
000188D8 ????                       tss.SS:         resw 1
000188DA ????                                   resw 1
000188DC ????                       tss.DS:         resw 1
000188DE ????                                   resw 1
000188E0 ????                       tss.FS:         resw 1
000188E2 ????                                   resw 1
000188E4 ????                       tss.GS:         resw 1
000188E6 ????                                   resw 1
000188E8 ????                       tss.LDTR:   resw 1
000188EA ????                                   resw 1
                                    ; tss offset 100
000188EC ????                                   resw 1
000188EE ????                       tss.IOPB:   resw 1
                                    ; tss offset 104
                                    tss_end:

000188F0 ????????                   k_page_dir:  resd 1 ; Kernel's (System) Page Directory address
                                                        ; (Physical address = Virtual address)
000188F4 ????????                   memory_size: resd 1 ; memory size in pages
000188F8 ????????                   free_pages:  resd 1 ; number of free pages
000188FC ????????                   next_page:   resd 1 ; offset value in M.A.T. for
                                                        ; first free page search
00018900 ????????                   last_page:   resd 1 ; offset value in M.A.T. which
                                                        ; next free page search will be
                                                        ; stopped after it. (end of M.A.T.)
00018904 ????????                   first_page:  resd 1 ; offset value in M.A.T. which
                                                        ; first free page search
                                                        ; will be started on it. (for user)
00018908 ????????                   mat_size:    resd 1 ; Memory Allocation Table size in pages

                                    ; 20/11/2020
                                    ;vbe2bios:    resw 1 ; VBE2 video bios ID (bochs/qemu)
                                    ;               ; (0B0C4h or 0B0C5h for bochs/plex86 vgabios)

                                    ; 02/09/2014 (Retro UNIX 386 v1)
                                    ; 04/12/2013 (Retro UNIX 8086 v1)
0001890C ????                       CRT_START:   resw 1        ; starting address in regen buffer
                                                       ; NOTE: active page only
0001890E <res 10h>                  CURSOR_POSN: resw 8 ; cursor positions for video pages
                                    ACTIVE_PAGE:
```

```
3580 0001891E ??                    ptty:            resb 1 ; current tty
3581                                ; 01/07/2015 - 29/01/2016
3582 0001891F ??                    ccolor:          resb 1 ; current color attribute
3583                                ; 26/10/2015
3584                                ; 07/09/2014
3585 00018920 <res 14h>             ttychr:      resw ntty+2 ; Character buffer (multiscreen)
3586
3587                                ; 18/05/2015 (03/06/2013 - Retro UNIX 8086 v1 feature only!)
3588 00018934 ????????              p_time:      resd 1     ; present time (for systime & sysmdate)
3589
3590                                ; 18/05/2015 (16/08/2013 - Retro UNIX 8086 v1 feature only !)
3591                                ; (open mode locks for pseudo TTYs)
3592                                ; [ major tty locks (return error in any conflicts) ]
3593 00018938 <res 14h>             ttyl:        resw ntty+2 ; opening locks for TTYs.
3594
3595                                ; 15/04/2015 (Retro UNIX 386 v1)
3596                                ; 22/09/2013 (Retro UNIX 8086 v1)
3597 0001894C <res Ah>              wlist:       resb ntty+2 ; wait channel list (0 to 9 for TTYs)
3598                                ; 15/04/2015 (Retro UNIX 386 v1)
3599                                ;; 12/07/2014 -> sp_init set comm. parameters as 0E3h
3600                                ;; 0 means serial port is not available
3601                                ;;comprm: ; 25/06/2014
3602 00018956 ??                    com1p:       resb 1  ;;0E3h
3603 00018957 ??                    com2p:       resb 1  ;;0E3h
3604
3605                                ; 17/11/2015
3606                                ; request for response (from the terminal)
3607 00018958 ????                  req_resp:    resw 1
3608                                ; 07/11/2015
3609 0001895A ??                    ccomport:    resb 1 ; current COM (serial) port
3610                                                    ; (0= COM1, 1= COM2)
3611                                ; 09/11/2015
3612 0001895B ??                    comqr:           resb 1 ; 'query or response' sign (u9.s, 'sndc')
3613                                ; 07/11/2015
3614 0001895C ????                  rchar:           resw 1 ; last received char for COM 1 and COM 2
3615 0001895E ????                  schar:           resw 1 ; last sent char for COM 1 and COM 2
3616
3617                                ; 22/08/2014 (RTC)
3618                                ; (Packed BCD)
3619 00018960 ??                    time_seconds: resb 1
3620 00018961 ??                    time_minutes: resb 1
3621 00018962 ??                    time_hours:   resb 1
3622 00018963 ??                    date_wday:    resb 1
3623 00018964 ??                    date_day:     resb 1
3624 00018965 ??                    date_month:   resb 1
3625 00018966 ??                    date_year:    resb 1
3626 00018967 ??                    date_century: resb 1
3627
3628                                ; 24/01/2016
3629 00018968 ????????              RTC_LH:            resd 1
3630 0001896C ??                    RTC_WAIT_FLAG: resb 1
3631 0001896D ??                    USER_FLAG:     resb 1
3632                                ; 19/05/2016
3633                                ;RTC_second:
3634 0001896E ??                    RTC_2Hz:       resb 1 ;  from 2Hz interrupt to 1Hz timer event function
3635
3636                                %include 'diskbss.s'     ; UNINITIALIZED DISK (BIOS) DATA
3637                              <1> ; ****************************************************************************
3638                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskbss.s
3639                              <1> ; ----------------------------------------------------------------------------
3640                              <1> ; Last Update: 24/01/2016 (11/04/2021)
3641                              <1> ; ----------------------------------------------------------------------------
3642                              <1> ; Beginning: 24/01/2016
3643                              <1> ; ----------------------------------------------------------------------------
3644                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
3645                              <1> ; ----------------------------------------------------------------------------
3646                              <1> ; Turkish Rational DOS
3647                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
3648                              <1> ;
3649                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3650                              <1> ; diskbss.inc (10/07/2015)
3651                              <1> ;
3652                              <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
3653                              <1> ; ****************************************************************************
3654                              <1>
3655                              <1> ; Retro UNIX 386 v1 Kernel - DISKBSS.INC
3656                              <1> ; Last Modification: 10/07/2015
3657                              <1> ;     (Unitialized Disk Parameters Data section for 'DISKIO.INC')
3658                              <1>
3659 0001896F ??                 <1> alignb 2
3660                              <1>
3661                              <1> ;----------------------------------------
3662                              <1> ;     TIMER DATA AREA           :
3663                              <1> ;----------------------------------------
3664                              <1>
3665                              <1> TIMER_LH:    ; 16/02/205
3666 00018970 ????               <1> TIMER_LOW:      resw     1               ; LOW WORD OF TIMER COUNT
3667 00018972 ????               <1> TIMER_HIGH:     resw     1               ; HIGH WORD OF TIMER COUNT
3668 00018974 ??                 <1> TIMER_OFL:      resb     1               ; TIMER HAS ROLLED OVER SINCE LAST READ
3669                              <1>
3670                              <1> ;----------------------------------------
3671                              <1> ;     DISKETTE DATA AREAS       :
3672                              <1> ;----------------------------------------
3673                              <1>
3674 00018975 ??                 <1> SEEK_STATUS: resb   1
3675 00018976 ??                 <1> MOTOR_STATUS:      resb   1
3676 00018977 ??                 <1> MOTOR_COUNT: resb   1
3677 00018978 ??                 <1> DSKETTE_STATUS:    resb   1
3678 00018979 ??????????????     <1> NEC_STATUS:  resb   7
3679                              <1>
3680                              <1> ;----------------------------------------
3681                              <1> ;     ADDITIONAL MEDIA DATA           :
3682                              <1> ;----------------------------------------
3683                              <1>
3684 00018980 ??                 <1> LASTRATE:    resb   1
```

```
3685 00018981 ??                <1> HF_STATUS:   resb   1
3686 00018982 ??                <1> HF_ERROR:    resb   1
3687 00018983 ??                <1> HF_INT_FLAG: resb   1
3688 00018984 ??                <1> HF_CNTRL:    resb   1
3689 00018985 ????????          <1> DSK_STATE:   resb   4
3690 00018989 ????              <1> DSK_TRK:     resb   2
3691                            <1>
3692                            <1> ;----------------------------------------
3693                            <1> ;     FIXED DISK DATA AREAS            :
3694                            <1> ;----------------------------------------
3695                            <1>
3696 0001898B ??                <1> DISK_STATUS1:      resb   1            ; FIXED DISK STATUS
3697 0001898C ??                <1> HF_NUM:            resb   1            ; COUNT OF FIXED DISK DRIVES
3698 0001898D ??                <1> CONTROL_BYTE:      resb   1            ; HEAD CONTROL BYTE
3699                            <1> ;@PORT_OFF   resb   1           ; RESERVED (PORT OFFSET)
3700                            <1> ;port1_off   resb   1           ; Hard disk controller 1 - port offset
3701                            <1> ;port2_off   resb   1           ; Hard disk controller 2 - port offset
3702                            <1>
3703 0001898E ????              <1> alignb 4
3704                            <1>
3705                            <1> ;HF_TBL_VEC: resd   1           ; Primary master disk param. tbl. pointer
3706                            <1> ;HF1_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
3707                            <1> HF_TBL_VEC: ; 22/12/2014
3708 00018990 ????????          <1> HDPM_TBL_VEC:      resd   1            ; Primary master disk param. tbl. pointer
3709 00018994 ????????          <1> HDPS_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
3710 00018998 ????????          <1> HDSM_TBL_VEC:      resd   1            ; Secondary master disk param. tbl. pointer
3711 0001899C ????????          <1> HDSS_TBL_VEC:      resd   1            ; Secondary slave disk param. tbl. pointer
3712                            <1>
3713                            <1> ; 03/01/2015
3714 000189A0 ??                <1> LBAMode:           resb   1
3715                            <1>
3716                            <1> ; ****************************************************************************
3637
3638                                ;;; Real Mode Data (10/07/2015 - BSS)
3639
3640                                ;alignb 2
3641
3642                                ; 10/01/2016
3643                                %include 'trdoskx.s'      ; UNINITIALIZED KERNEL (Logical Drive & FS) DATA
3644                            <1> ; ****************************************************************************
3645                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.2) - UNINITIALIZED DATA : trdoskx.s
3646                            <1> ; ----------------------------------------------------------------------------
3647                            <1> ; Last Update: 30/08/2020
3648                            <1> ; ----------------------------------------------------------------------------
3649                            <1> ; Beginning: 04/01/2016
3650                            <1> ; ----------------------------------------------------------------------------
3651                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3652                            <1> ; ----------------------------------------------------------------------------
3653                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3654                            <1> ; TRDOS2.ASM (09/11/2011)
3655                            <1> ; ****************************************************************************
3656                            <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
3657                            <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
3658                            <1> ; DIR.ASM      [17/01/2004] Last Update: 09/10/2011
3659                            <1> ; CMD_INTR.ASM [29/01/2005] Last update: 09/11/2011
3660                            <1> ; DRV_FAT.ASM  [07/07/2009] Last update: 21/08/2011
3661                            <1>
3662 000189A1 ??????            <1> alignb 4
3663                            <1>
3664                            <1> ; MAINPROG.ASM
3665 000189A4 ????????          <1> MainProgCfg_FileSize:   resd 1 ; 14/04/2016
3666 000189A8 ????????          <1> MainProgCfg_LineOffset: resd 1 ; 14/04/2016
3667                            <1>
3668 000189AC ????????          <1> Current_VolSerial: resd 1
3669                            <1>
3670 000189B0 ????????          <1> Current_Dir_FCluster: resd 1
3671                            <1>
3672 000189B4 ??                <1> Current_Dir_Level: resb 1
3673 000189B5 ??                <1> Current_FATType: resb 1
3674 000189B6 ??                <1> Current_Drv: resb 1
3675 000189B7 ??                <1> Current_Dir_Drv:   resb 1 ; '?'
3676 000189B8 ??                <1>                    resb 1 ; ':'
3677 000189B9 ??                <1> Current_Dir_Root:  resb 1 ; '/'
3678 000189BA <res 5Ah>          <1> Current_Directory: resb 90
3679 00018A14 ??                <1> End_Of_Current_Dir_Str: resb 1
3680 00018A15 ??                <1> Current_Dir_StrLen: resb 1
3681                            <1>
3682 00018A16 ??                <1> CursorColumn:      resb 1
3683 00018A17 ??                <1> CmdArgStart:       resb 1
3684                            <1>
3685                            <1> ; 03/02/2016
3686 00018A18 <res 4Eh>          <1> Remark:     resb 78
3687                            <1>
3688 00018A66 <res 50h>          <1> CommandBuffer: resb 80
3689                            <1>
3690 00018AB6 <res 100h>         <1> TextBuffer:  resb 256
3691                            <1>
3692                            <1> MasterBootBuff:
3693 00018BB6 <res 1BEh>         <1> MasterBootCode: resb 1BEh
3694 00018D74 <res 40h>          <1> PartitionTable: resb 64
3695 00018DB4 ????              <1> MBIDCode: resw 1
3696                            <1>
3697                            <1> PTable_Buffer:
3698 00018DB6 <res 40h>          <1> PTable_hd0: resb 64
3699 00018DF6 <res 40h>          <1> PTable_hd1: resb 64
3700 00018E36 <res 40h>          <1> PTable_hd2: resb 64
3701 00018E76 <res 40h>          <1> PTable_hd3: resb 64
3702                            <1> ; 15/07/2020
3703                            <1> ;PTable_ep0: resb 64
3704                            <1> ;PTable_ep1: resb 64
3705                            <1> ;PTable_ep2: resb 64
3706                            <1> ;PTable_ep3: resb 64
3707                            <1>
3708                            <1> ; 13/08/2020
3709 00018EB6 ??                <1> scount: resb 1 ; 16/05/2016 (diskio.s, 'int33h:')
```

```
3710 00018EB7 ??                <1>       resb 1
3711 00018EB8 ??                <1>       resb 1
3712 00018EB9 ??                <1>       resb 1
3713                            <1>
3714 00018EBA ??                <1> HD_LBA_yes: resb 1
3715 00018EBB ??                <1> PP_Counter: resb 1
3716 00018EBC ??                <1> EP_Counter: resb 1
3717                            <1> ; 13/08/2020
3718 00018EBD ??                <1> LD_Counter: resb 1
3719                            <1>
3720                            <1> ; 30/08/2020
3721 00018EBE ????????          <1> MBR_EP_StartSector: resd 1
3722                            <1>             ; Extd partition start sector as in MBR
3723 00018EC2 ????????          <1> EP_StartSector: resd 1 ; next extd partition start sector
3724                            <1>       ; 15/07/2020
3725                            <1>       ;resd 1
3726                            <1>       ;resd 1
3727                            <1>
3728                            <1> ; 20/07/2020
3729 00018EC6 <res 200h>        <1> DOSBootSectorBuff: resb 512
3730                            <1> ; 15/07/2020
3731                            <1> ;DOSBootSectorBuff: resb 446 ; 1BEh
3732                            <1> ;MiniPartitionTable: resb 64 ;  40h
3733                            <1> ;MiniPartitionMagic: resw  1 ;  02h
3734                            <1>
3735                            <1> FAT_BuffDescriptor:
3736 000190C6 ????????          <1> FAT_CurrentCluster: resd 1
3737 000190CA ??                <1> FAT_BuffValidData: resb 1
3738 000190CB ??                <1> FAT_BuffDrvName: resb 1
3739 000190CC ????              <1> FAT_BuffOffset: resw 1
3740 000190CE ????????          <1> FAT_BuffSector: resd 1
3741                            <1>
3742 000190D2 ????????          <1> FAT_ClusterCounter: resd 1
3743 000190D6 ????????          <1> LastCluster: resd 1
3744                            <1>
3745                            <1> ; 16/05/2016
3746                            <1> ;; 18/03/2016 (TRDOS v2.0)
3747                            <1> ;ClusterBuffer_Valid: resb 1
3748                            <1>
3749                            <1> Dir_BuffDescriptor:
3750 000190DA ??                <1> DirBuff_DRV: resb 1
3751 000190DB ??                <1> DirBuff_FATType: resb 1
3752 000190DC ??                <1> DirBuff_ValidData: resb 1
3753 000190DD ????              <1> DirBuff_CurrentEntry: resw 1
3754 000190DF ????              <1> DirBuff_LastEntry: resw 1
3755 000190E1 ????????          <1> DirBuff_Cluster: resd 1
3756 000190E5 ????              <1> DirBuffer_Size: resw 1
3757                            <1> ;DirBuff_EntryCounter: resw 1
3758                            <1>
3759                            <1> ; 01/02/2016
3760                            <1> ; these are on (real mode) segment 8000h and later
3761                            <1> ; FAT_Buffer:      resb 1536 ; 3 sectors
3762                            <1> ; Dir_Buffer:      resb 512*32
3763                            <1> ; Logical_DOSDisks: resb 6656 ; 26 * 256 bytes
3764                            <1>
3765                            <1> ; 18/01/2016
3766                            <1>
3767 000190E7 ????????          <1> FreeClusterCount: resd 1
3768                            <1>
3769 000190EB ????????          <1> VolSize_Unit1:   resd 1
3770 000190EF ????????          <1> VolSize_Unit2:   resd 1
3771                            <1>
3772 000190F3 ????????          <1> Vol_Tot_Sec_Str_Start: resd 1
3773 000190F7 <res Ah>          <1> Vol_Tot_Sec_Str:       resb 10
3774 00019101 ??                <1> Vol_Tot_Sec_Str_End:   resb 1
3775 00019102 ??                <1> resb 1
3776 00019103 ????????          <1> Vol_Free_Sectors_Str_Start: resd 1
3777 00019107 <res Ah>          <1> Vol_Free_Sectors_Str:      resb 10
3778 00019111 ??                <1> Vol_Free_Sectors_Str_End:   resb 1
3779                            <1>
3780                            <1> ; 10/02/2016
3781 00019112 ??                <1> RUN_CDRV: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
3782                            <1>
3783                            <1> ; 24/01/2016
3784 00019113 <res 80h>         <1> PATH_Array:     resb 128 ; DIR.ASM ; 09/10/2011
3785                            <1> ; 06/02/2016
3786 00019193 ????????          <1> CCD_DriveDT: resd 1 ; DIR.ASM ; (word)
3787 00019197 ??                <1> CCD_Level:   resb 1 ; DIR.ASM
3788 00019198 ??                <1> Last_Dir_Level:    resb 1 ; DIR.ASM
3789                            <1> ;
3790 00019199 ????              <1> CDLF_AttributesMask: resw 1 ; DIR.ASM
3791 0001919B ????????          <1> CDLF_FNAddress:    resd 1 ; DIR.ASM (word)
3792 0001919F ????              <1> CDLF_DEType: resw 1 ; DIR.ASM
3793                            <1> ;
3794 000191A1 ??                <1> CD_COMMAND:  resb 1 ; DIR.ASM
3795                            <1>
3796 000191A2 ????              <1> alignb 4
3797                            <1>
3798                            <1> ; 29/01/2016
3799 000191A4 ??                <1> Program_Exit: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
3800                            <1>
3801                            <1> ;alignb 4
3802                            <1> ; 23/02/2016
3803 000191A5 ??                <1> disk_rw_op:  resb 1 ;  0 = disk read, 1 = disk write
3804                            <1> ;disk_rw_spt:      resb 1 ; sectors per track (<= 63) /// (<256)
3805                            <1> ; 31/01/2016
3806 000191A6 ??                <1> retry_count:       resb 1 ; DISK_IO.ASM ; 20/07/2011 (CHS_RetryCount)
3807 000191A7 ??                <1> disk_rw_err:       resb 1 ; DISK_IO.ASM ; (Disk_IO_err_code)
3808 000191A8 ????????          <1> sector_count:      resd 1 ; DISK_IO.ASM ; (Disk_RW_SectorCount)
3809                            <1>
3810                            <1> ; 06/02/2016 (long name)
3811 000191AC ????              <1> FDE_AttrMask:      resw 1 ; DIR.ASM
3812 000191AE ????              <1> AmbiguousFileName: resw 1 ; DIR.ASM
3813 000191B0 ??                <1> PreviousAttr:      resb 1 ; DIR.ASM
3814                            <1> ;
```

```
3815 000191B1 ??               <1> LongNameFound:   resb 1     ; DIR.ASM
3816 000191B2 ??               <1> LFN_EntryLength: resb 1   ; DIR.ASM
3817 000191B3 ??               <1> LFN_CheckSum:    resb 1   ; DIR.ASM
3818 000191B4 <res 84h>        <1> LongFileName:    resb 132 ; DIR.ASM
3819                           <1>
3820                           <1> ;PATH_Array_Ptr: resw 1 ; DIR.ASM
3821 00019238 ??               <1> PATH_CDLevel: resb 1 ; DIR.ASM
3822 00019239 ??               <1> PATH_Level:   resb 1 ; DIR.ASM
3823                           <1>
3824                           <1> ; 07/02/2016
3825 0001923A <res Dh>         <1> Dir_File_Name: resb 13 ; DIR.ASM ; 09/10/2011
3826                           <1>
3827                           <1> ; 10/02/2016
3828 00019247 <res Dh>         <1> Dir_Entry_Name: resb 13 ; DIR.ASM
3829                           <1>
3830                           <1> alignb 2
3831                           <1>
3832 00019254 ????             <1> AttributesMask: resw 1 ; CMD_INTR.ASM ; 09/11/2011
3833                           <1>
3834                           <1> ; 10/02/2016 (128 bytes -> 126 bytes)
3835                           <1> ; 08/02/2016
3836                           <1> ;FFF Structure (128 bytes) ; DIR.ASM ; 09/10/2011
3837 00019256 ??               <1> FindFile_Drv:            resb 1
3838 00019257 <res 41h>        <1> FindFile_Directory:      resb 65
3839 00019298 <res Dh>         <1> FindFile_Name:           resb 13
3840                           <1> FindFile_LongNameEntryLength:
3841 000192A5 ??               <1> FindFile_LongNameYes:    resb 1 ; Sign for longname procedures
3842                           <1> ;Above 80 bytes form
3843                           <1> ;TR-DOS Source/Destination File FullName Format/Structure
3844 000192A6 ????             <1> FindFile_AttributesMask:  resw 1
3845 000192A8 <res 20h>        <1> FindFile_DirEntry:       resb 32
3846 000192C8 ????????         <1> FindFile_DirFirstCluster: resd 1
3847 000192CC ????????         <1> FindFile_DirCluster:     resd 1
3848 000192D0 ????             <1> FindFile_DirEntryNumber:  resw 1
3849 000192D2 ????             <1> FindFile_MatchCounter:    resw 1
3850 000192D4 ????             <1> FindFile_Reserved:        resw 1 ; 06/03/2016
3851                           <1>
3852 000192D6 ????????         <1> First_Path_Pos: resd 1    ; DIR.ASM ; 09/10/2011
3853 000192DA ????????         <1> Last_Slash_Pos: resd 1    ; DIR.ASM
3854                           <1>
3855                           <1> ; 10/02/2016
3856 000192DE ????             <1> File_Count:     resw 1    ; DIR.ASM ; 09/10/2011
3857 000192E0 ????             <1> Dir_Count:      resw 1
3858 000192E2 ????????         <1> Total_FSize:    resd 1
3859 000192E6 ????????         <1> TFS_Dec_Begin:  resd 1
3860 000192EA <res Ah>         <1>                 resb 10
3861 000192F4 ??               <1> TFS_Dec_End:    resb 1
3862                           <1>
3863 000192F5 ??               <1> PrintDir_RowCounter: resb 1
3864                           <1>
3865 000192F6 ????             <1> alignb 4
3866                           <1> ; 15/02/2015 ('show' command variables)
3867 000192F8 ????????         <1> Show_FDT:    resd 1
3868 000192FC ????????         <1> Show_LDDDT:  resd 1
3869 00019300 ????????         <1> Show_Cluster: resd 1
3870 00019304 ????????         <1> Show_FileSize: resd 1
3871 00019308 ????????         <1> Show_FilePointer: resd 1
3872 0001930C ????             <1> Show_ClusterPointer: resw 1
3873 0001930E ????             <1> Show_ClusterSize: resw 1
3874 00019310 ??               <1> Show_RowCount: resb 1
3875                           <1>
3876 00019311 ??????           <1> alignb 4
3877                           <1> ; 21/02/2016
3878 00019314 ????????         <1> DelFile_FNPointer: resd 1 ; ; CMD_INTR.ASM (word) ; 09/11/2011
3879                           <1> ; 27/02/2016
3880                           <1> ; DIR.ASM (09/10/2011)
3881 00019318 ????????         <1> DelFile_FCluster:  resd 1
3882 0001931C ????             <1> DelFile_EntryCounter: resw 1
3883 0001931E ??               <1> DelFile_LNEL:      resb 1
3884 0001931F ??               <1> resb 1
3885                           <1>
3886                           <1> ; DIR.ASM
3887 00019320 ????????         <1> mkdir_DirName_Offset: resd 1
3888 00019324 ????????         <1> mkdir_FFCluster:   resd 1
3889 00019328 ????????         <1> mkdir_LastDirCluster: resd 1
3890 0001932C ????????         <1> mkdir_FreeSectors: resd 1
3891 00019330 ????             <1> mkdir_attrib:      resw 1
3892 00019332 ??               <1> mkdir_SecPerClust: resb 1
3893 00019333 ??               <1> mkdir_add_new_cluster: resb 1
3894 00019334 <res Dh>         <1> mkdir_Name:        resb 13
3895 00019341 ????             <1> resw 1 ; 01/03/2016
3896                           <1> ; 27/02/2016
3897 00019343 ??               <1> RmDir_MultiClusters:  resb 1
3898 00019344 ????????         <1> RmDir_DirEntryOffset: resd 1 ; 01/03/2016 (word -> dword)
3899 00019348 ????????         <1> RmDir_ParentDirCluster: resd 1
3900 0001934C ????????         <1> RmDir_DirLastCluster:   resd 1
3901 00019350 ????????         <1> RmDir_PreviousCluster:  resd 1
3902                           <1> ; 22/02/2016
3903 00019354 ??               <1> UPDLMDT_CDirLevel: resb 1
3904 00019355 ????????         <1> UPDLMDT_CDirFCluster: resd 1
3905                           <1>
3906 00019359 ??????           <1> alignb 4
3907                           <1> ; DRV_FAT.ASM ; 21/08/2011
3908 0001935C ????????         <1> gffc_next_free_cluster:  resd 1
3909 00019360 ????????         <1> gffc_first_free_cluster: resd 1
3910 00019364 ????????         <1> gffc_last_free_cluster:  resd 1
3911                           <1>
3912                           <1> ;29/04/2016
3913                           <1> Cluster_Index: ; resd 1
3914                           <1> ; 22/02/2016
3915 00019368 ????????         <1> ClusterValue: resd 1
3916                           <1> ; 04/03/2016
3917 0001936C ??               <1> Attributes: resb 1
3918                           <1> ;;CFS_error: resb 1 ;; 01/03/2016
3919 0001936D ??               <1> resb 1
```

```
3920 0001936E ??                <1> CFS_OPType: resb 1
3921 0001936F ??                <1> CFS_Drv:    resb 1
3922 00019370 ????????          <1> CFS_CC:     resd 1
3923 00019374 ????????          <1> CFS_FAT32FSINFOSEC: resd 1
3924 00019378 ????????          <1> CFS_FAT32FC: resd 1
3925                            <1>
3926                            <1> ; 27/02/2016
3927                            <1> ;alignb 4
3928 0001937C ????????          <1> glc_prevcluster: resd 1 ; DRV_FAT.ASM (21/08/2011)
3929                            <1> ; 22/10/2016
3930 00019380 ????????          <1> glc_index:   resd 1 ;  Last Cluster Index (22/10/2016)
3931                            <1>
3932                            <1> ; DIR.ASM
3933 00019384 ????              <1> DLN_EntryNumber: resw 1
3934 00019386 ??                <1> DLN_40h:     resb 1
3935                            <1> ; 28/02/2016
3936 00019387 ??                <1> TCC_FATErr:  resb 1 ; DRV_FAT.ASM
3937                            <1>
3938                            <1> alignb 4
3939                            <1> ; DIR.ASM (09/10/2011)
3940 00019388 ????              <1> LCDE_EntryIndex: resw 1 ; LCDE_EntryOffset
3941 0001938A ????              <1> LCDE_ClusterSN:  resw 1
3942 0001938C ????????          <1> LCDE_Cluster:    resd 1
3943 00019390 ????????          <1> LCDE_ByteOffset: resd 1
3944                            <1>
3945                            <1> ;alignb4
3946                            <1> ; 06/03/2016 (word -> dword)
3947                            <1> ; CMD_INTR.ASM (01/08/2010)
3948 00019394 ????????          <1> SourceFilePath:      resd 1
3949 00019398 ????????          <1> DestinationFilePath: resd 1
3950                            <1>
3951                            <1> ;alignb 4
3952                            <1> ; 06/03/2016
3953                            <1> ; FILE.ASM (09/10/2011)
3954                            <1> ;Source File Structure (same with 'Find File' Structure)
3955 0001939C ??                <1> SourceFile_Drv:                 resb 1
3956 0001939D <res 41h>         <1> SourceFile_Directory:           resb 65
3957 000193DE <res Dh>          <1> SourceFile_Name:          resb 13
3958                            <1> SourceFile_LongNameEntryLength:
3959 000193EB ??                <1> SourceFile_LongNameYes:         resb 1 ; Sign for longname procedures
3960                            <1> ;Above 80 bytes
3961                            <1> ;is TR-DOS Source File FullName Format/Structure
3962 000193EC ????              <1> SourceFile_AttributesMask:      resw 1
3963 000193EE <res 20h>         <1> SourceFile_DirEntry:            resb 32
3964 0001940E ????????          <1> SourceFile_DirFirstCluster:     resd 1
3965 00019412 ????????          <1> SourceFile_DirCluster:          resd 1
3966 00019416 ????              <1> SourceFile_DirEntryNumber:      resw 1
3967 00019418 ????              <1> SourceFile_MatchCounter:  resw 1
3968                            <1> ; 16/03/2016
3969 0001941A ??                <1> SourceFile_SecPerClust:         resb 1
3970 0001941B ??                <1> SourceFile_Reserved:            resb 1
3971                            <1> ; Above is 128 bytes
3972                            <1>
3973                            <1> ;Destination File Structure (same with 'Find File' Structure)
3974 0001941C ??                <1> DestinationFile_Drv:            resb 1
3975 0001941D <res 41h>         <1> DestinationFile_Directory:      resb 65
3976 0001945E <res Dh>          <1> DestinationFile_Name:           resb 13
3977                            <1> DestinationFile_LongNameEntryLength:
3978 0001946B ??                <1> DestinationFile_LongNameYes:    resb 1 ; Sign for longname procedures
3979                            <1> ;Above 80 bytes
3980                            <1> ;is TR-DOS Destination File FullName Format/Structure
3981 0001946C ????              <1> DestinationFile_AttributesMask: resw 1
3982 0001946E <res 20h>         <1> DestinationFile_DirEntry: resb 32
3983 0001948E ????????          <1> DestinationFile_DirFirstCluster: resd 1
3984 00019492 ????????          <1> DestinationFile_DirCluster:     resd 1
3985 00019496 ????              <1> DestinationFile_DirEntryNumber: resw 1
3986 00019498 ????              <1> DestinationFile_MatchCounter:   resw 1
3987                            <1> ; 16/03/2016
3988 0001949A ??                <1> DestinationFile_SecPerClust:    resb 1
3989 0001949B ??                <1> DestinationFile_Reserved: resb 1
3990                            <1> ; Above is 128 bytes
3991                            <1>
3992                            <1> ; 24/04/2016
3993 0001949C ????              <1> resw 1
3994                            <1>
3995                            <1> ; 10/03/2016
3996                            <1> ; FILE.ASM
3997 0001949E ??                <1> move_cmd_phase:    resb 1
3998 0001949F ??                <1> msftdf_sf_df_drv:  resb 1
3999 000194A0 ????????          <1> msftdf_drv_offset: resd 1
4000                            <1>
4001                            <1> ; 11/03/2016
4002                            <1> ; DRV_FAT.ASM (21/08/2011)
4003 000194A4 ????????          <1> FAT_anc_LCluster:  resd 1
4004 000194A8 ????????          <1> FAT_anc_FFCluster: resd 1
4005                            <1>
4006                            <1> ;alignb 4
4007                            <1>
4008                            <1> ; 14/03/2016
4009                            <1> ; TRDOS 386 = TRDOS v2.0 feature only !
4010                            <1> ; 'allocate_memory_block' in 'memory.s'
4011 000194AC ????????          <1> mem_ipg_count:     resd 1 ; page count (for contiguous allocation)
4012 000194B0 ????????          <1> mem_pg_count:      resd 1 ; page count (for count down)
4013 000194B4 ????????          <1> mem_aperture:      resd 1 ; contiguous free pages (current)
4014 000194B8 ????????          <1> mem_max_aperture: resd 1 ; maximum value of contiguous free pages
4015 000194BC ????????          <1> mem_pg_pos:  resd 1 ; mem. position (page #) of current aperture
4016 000194C0 ????????          <1> mem_max_pg_pos: resd 1 ; mem. position (page #) of max. aperture
4017                            <1>
4018                            <1> ; 15/03/2016
4019                            <1> ; FILE.ASM ('copy_source_file_to_destination_file')
4020 000194C4 ??                <1> copy_cmd_phase:       resb 1
4021 000194C5 ??                <1> csftdf_rw_err:        resb 1
4022 000194C6 ??                <1> DestinationFileFound: resb 1
4023 000194C7 ??                <1> csftdf_cdrv:          resb 1
4024 000194C8 ????????          <1> csftdf_filesize:      resd 1
```

```
4025                            <1> ; TRDOS386 (TRDOS v2.0)
4026 000194CC ????????          <1> csftdf_sf_mem_addr:   resd 1
4027 000194D0 ????????          <1> csftdf_sf_mem_bsize:  resd 1
4028                            <1> ;
4029                            <1>
4030 000194D4 ????????          <1> csftdf_sf_cluster:    resd 1 ; 16/03/2016
4031 000194D8 ????????          <1> csftdf_df_cluster:    resd 1
4032                            <1> ; 16/03/2016
4033 000194DC ????????          <1> csftdf_r_size:        resd 1
4034 000194E0 ????????          <1> csftdf_w_size:        resd 1
4035 000194E4 ????????          <1> csftdf_sf_rbytes:     resd 1
4036 000194E8 ????????          <1> csftdf_df_wbytes:     resd 1
4037 000194EC ??                <1> csftdf_percentage:    resb 1
4038                            <1> ; 17/03/2016
4039 000194ED ??                <1> csftdf_videopage:     resb 1
4040 000194EE ????              <1> csftdf_cursorpos:     resw 1
4041 000194F0 ????????          <1> csftdf_sf_drv_dt:     resd 1
4042 000194F4 ????????          <1> csftdf_df_drv_dt:     resd 1
4043                            <1>
4044                            <1> ; 21/03/2016
4045                            <1> ; 20/03/2016
4046                            <1> ; FILE.ASM
4047 000194F8 ????????          <1> createfile_Name_Offset:  resd 1
4048 000194FC ????????          <1> createfile_FreeSectors:  resd 1
4049 00019500 ????????          <1> createfile_size:         resd 1
4050 00019504 ????????          <1> createfile_FFCluster:    resd 1 ; 11/03/2016
4051 00019508 ????????          <1> createfile_LastDirCluster: resd 1
4052 0001950C ????????          <1> createfile_Cluster:      resd 1
4053 00019510 ????????          <1> createfile_PCluster:     resd 1
4054 00019514 ??                <1> createfile_attrib:       resb 1
4055 00019515 ??                <1> createfile_SecPerClust:  resb 1
4056 00019516 ????              <1> createfile_DirIndex:     resw 1
4057 00019518 ????????          <1> createfile_CCount:       resd 1
4058 0001951C ????              <1> createfile_BytesPerSec:  resw 1 ; 23/03/2016
4059 0001951E ??                <1> createfile_wfc:          resb 1
4060 0001951F ??                <1> createfile_UpdatePDir:   resb 1 ; 31/03/2016
4061                            <1>
4062                            <1> ;alignb 4
4063                            <1>
4064                            <1> ; 11/04/2016
4065 00019520 ????              <1> env_var_length: resw 1
4066                            <1>
4067 00019522 ????              <1> alignb 4
4068                            <1>
4069                            <1> ; 25/04/2016
4070 00019524 ??                <1> readi.valid: resb 1 ; valid data (>0 = valid for readi)
4071 00019525 ??                <1> readi.drv:  resb 1 ; drive number (0, 1,2,3,4..)
4072 00019526 ??                <1> readi.spc:  resb 1 ; sectors per cluster for 'readi' drive
4073 00019527 ??                <1> readi.s_index:  resb 1 ; sector index in current cluster (buffer)
4074 00019528 ????????          <1> readi.sector: resd 1 ; current disk sector
4075 0001952C ????              <1> readi.bpc:  resw 1 ; bytes per cluster - 1
4076 0001952E ????              <1> readi.offset: resw 1 ; byte offset in cluster buffer
4077 00019530 ????????          <1> readi.cluster: resd 1 ; current cluster number
4078 00019534 ????????          <1> readi.c_index: resd 1 ; cluster index of the current cluster (0,1,2,3..)
4079 00019538 ????????          <1> readi.fclust: resd 1 ; first cluster of the current cluster
4080 0001953C ????????          <1> readi.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
4081                            <1> ;readi.buffer: resd 1 ; readi sector buffer address
4082                            <1>
4083                            <1> ;alignb 4
4084                            <1>
4085 00019540 ??                <1> writei.valid: resb 1 ; valid data (>0 = valid for writei)
4086 00019541 ??                <1> writei.drv:  resb 1 ; drive number (0, 1,2,3,4..)
4087 00019542 ??                <1> writei.spc:  resb 1 ; sectors per cluster for 'writei' drive
4088 00019543 ??                <1> writei.s_index: resb 1 ; sector index in current cluster (buffer)
4089 00019544 ????????          <1> writei.sector: resd 1 ; current disk sector
4090 00019548 ????              <1> writei.bpc:  resw 1 ; bytes per cluster - 1
4091 0001954A ????              <1> writei.offset: resw 1 ; byte offset in cluster buffer
4092 0001954C ????????          <1> writei.cluster: resd 1 ; current cluster number
4093 00019550 ????????          <1> writei.c_index: resd 1 ; cluster index of the current cluster (0,1,2,3..)
4094 00019554 ????????          <1> writei.fclust:  resd 1 ; first cluster of the current cluster
4095 00019558 ????????          <1> writei.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
4096                            <1> ;writei.buffer: resd 1 ; writei sector buffer address
4097 0001955C ????????          <1> writei.lclust:  resd 1 ; writei last cluster (mget_w) ; 23/10/2016
4098 00019560 ????????          <1> writei.l_index: resd 1 ; writei last cluster index (mget_w) ; 23/10/2016
4099 00019564 ??                <1> writei.ofn:  resb 1 ; open file number (to be written) ; 23/10/2016
4100                            <1>
4101 00019565 ??????            <1> alignb 4
4102                            <1>
4103                            <1> ; 29/04/2016
4104 00019568 ????????          <1> Run_CDirFC:  resd 1
4105 0001956C ??                <1> Run_Auto_Path: resb 1
4106 0001956D ??                <1> Run_Manual_Path: resb 1 ; 0 -> auto path sequence needed
4107 0001956E ??                <1> EXE_ID:      resb 1
4108 0001956F ??                <1> EXE_dot:     resb 1
4109                            <1>
4110                            <1> ; 06/05/2016
4111 00019570 ????????          <1> mainprog_return_addr: resd 1
4112 00019574 ????????          <1> last_error:  resd 1  ; this will be used to return error code to MainProg
4113                            <1>                      ; 'lasterror' keyword will be used later to get the
4114                            <1>                      ; last error code/number/status.
4115                            <1> ; 12/05/2016
4116 00019578 ????????          <1> video_eax:   resd 1  ; eax return value of video function
4117                            <1>
4118                            <1> ; 01/06/2016
4119 0001957C ????????          <1> user_buffer: resd 1  ; 'diskio.s' (INT 33h, Function 08h, floppy disk type)
4120                            <1>
4121                            <1> ; 21/05/2016 - TRDOS 386 ('swap/switch', 'rswap', [u.pri])
4122 00019580 ??                <1> priority:    resb 1  ; running priority level of process (0,1,2)
4123                            <1>                      ; (run queue which is process comes from)
4124                            <1> ; 22/05/2016 - TRDOS 386 ('set_run_sequence', 'rtc_int', 'u_timer')
4125 00019581 ??                <1> p_change:    resb 1  ; process change status (for timer events)
4126                            <1> ; 23/05/2016 - TRDOS 386 ('clock')
4127 00019582 ??                <1> multi_tasking: resb 1 ; Multi Tasking status (0 = disabled, >0 = enabled)
4128                            <1>                      ; (EBX will return with user buffer addr or disk type)
4129                            <1> ; 07/06/2016
```

```
4130 00019583 ??                <1> timer_events: resb 1  ; number of (active) timer events, <= 16
4131                            <1>
4132                            <1> ; 24/06/2016
4133 00019584 ??                <1> w_str_cmd:   resb 1 ; WRITE_STRING command (0,1,2,3) ; video.s
4134 00019585 ??                <1> p_crt_mode:  resb 1  ; previous video mode (=3 or 0), backup mark/sign
4135                            <1> ; 26/06/2016
4136 00019586 ??                <1> p_crt_page:  resb 1  ; previous active page (for 'set_mode')
4137                            <1> ; 04/07/2016
4138 00019587 ??                <1> noclearmem:  resb 1  ; if set, 'SET MODE' (INT 31h) function (AH = 4)
4139                            <1>                      ; will not clear the video memory
4140                            <1>                      ; (usable for graphics modes only)
4141                            <1> alignb 2
4142 00019588 ????              <1> CRT_LEN:     resw 1  ; length of regen buffer in bytes
4143 0001958A <res 10h>         <1> cursor_pposn:      resw 8  ; cursor positions backup
4144                            <1>
4145                            <1> ; 10/07/2016 ('VGA_FONT_SETUP', INT 43H address for x86 real mode bios)
4146 0001959A ????????          <1> VGA_INT43H:  resd 1 ; 0 = default (not configured by user)
4147                            <1>                      ; 0FFFFFFFFh = user defined fonts
4148                            <1>                      ; address:
4149                            <1>                      ;      vgafont8
4150                            <1>                      ;      vgafont16
4151                            <1>                      ;      vgafont14
4152                            <1>
4153                            <1> ; 25/07/2016
4154 0001959E ??                <1> VGA_MTYPE:   resb 1  ; 0=CTEXT,1=MTEXT,2=CGA,3=PLANAR1,4=PLANAR4,5=LINEAR
4155                            <1>
4156                            <1> ; 23/10/2016
4157 0001959F ??                <1> setfmod:     resb 1 ; update last modification date&time sign (if >0)
4158                            <1>                      ; (it is Open File Number + 1, if > 0)
4159                            <1> alignb 4
4160                            <1>
4161                            <1> ; 16/10/2016
4162 000195A0 ????????          <1> FFF_UBuffer: resd 1  ; User's buffer address for FFF & FNF system calls
4163                            <1> ; 15/10/2016
4164 000195A4 ??                <1> FFF_Valid:   resb 1  ; Find First File Structure validation byte
4165                            <1>                      ; 0  = invalid (Find Next File can't use FFF struct)
4166                            <1>                      ; >0 = valid, return type for FFF and Find Next File
4167                            <1>                      ; 24 = basic parameters, 24 bytes
4168                            <1>                      ; 128 = entire FFF structure/table, 128 bytes
4169                            <1> ; 16/10/2016 (FFF_Attrib: resw 1)
4170 000195A5 ??                <1> FFF_Attrib:  resb 1 ; Find First File attributes for Find Next File (LB)
4171 000195A6 ??                <1> FFF_RType:   resb 1  ; FFF return type (0 = Basic, >0 = complete) (HB)
4172                            <1> ; 16/10/2016 - 05/10/2016 (Set Working Path)
4173 000195A7 ??                <1> SWP_inv_fname:     resb 1 ; Set Working Path - Invalid File Name
4174 000195A8 ????              <1> SWP_Mode:    resw 1 ; Set Working Path - Mode
4175 000195AA ??                <1> SWP_DRV:     resb 1 ; Set Working Path - Drive
4176 000195AB ??                <1> SWP_DRV_chg: resb 1 ; Set Working Path - Drive Change
4177                            <1>
4178                            <1> ; 27/02/2017
4179 000195AC ??                <1> fpready:     resb 1 ; '80387 fpu is ready' flag
4180                            <1>
4181                            <1> ; 08/10/2016
4182 000195AD <res 9h>          <1> device_name: resb 9  ; capitalized (and zero padded) device canem
4183                            <1>                      ; (example: "TTY0",0,0,0,0,0")
4184                            <1>
4185 000195B6 ????              <1> alignb 4
4186                            <1>
4187                            <1> ; 08/10/2016
4188                            <1> ; 07/10/2016
4189                            <1> ; Table of kernel devices (which do not use installable device drivers)
4190                            <1> ; has been coded into KERNEL (trdosk9.s)
4191                            <1> ; 07/10/2016
4192                            <1> ; 8 installable device drivers available to install (NUMIDEV)
4193 000195B8 <res 20h>          <1> IDEV_PGDIR: resd NUMIDEV
4194                            <1>                      ; Page directories of installable device drivers
4195                            <1>                      ;
4196                            <1>                      ; Note: Virtual start address is always 400000h
4197                            <1>                      ; (end of the 1st 4MB). [org 400000h]
4198                            <1>                      ; Segments: KCODE, KDATA
4199                            <1>                      ; Method: call 400000h (after changing page dir)
4200                            <1>                      ; Query code located at the start (400000h).
4201                            <1>                      ; Query code returns with
4202                            <1>                      ;   eax = device type and driver version
4203                            <1>                      ;         AL = Device Type minor
4204                            <1>                      ;         AH = Device Type major
4205                            <1>                      ;         Byte 16-23 : Version minor
4206                            <1>                      ;        Byte 24-31 : Version major - 1
4207                            <1>                      ;                     (0:0 -> 1.0)
4208                            <1>                      ;   ebx = initialization code address
4209                            <1>                      ;   ecx = configuration table address
4210                            <1>                      ;   edx = description table address
4211                            <1>                      ;   esi = device (default) name address (ASCIIZ)
4212                            <1>                      ;       (name has "/DEV/" prefix)
4213                            <1>                      ;   edi = dispatch table address
4214                            <1>                      ;        (for calling kernel-device functions)
4215                            <1>                      ;   ebp = address table address
4216                            <1>                      ; Initialization code returns with
4217                            <1>                      ;   eax = open code address
4218                            <1>                      ;   ecx = close code address
4219                            <1>                      ;   ebx = read code address
4220                            <1>                      ;   edx = write code address
4221                            <1>                      ;   esi = IOCTL code address
4222                            <1>                      ;   edi = dispatch table address
4223                            <1>                      ;   ebp = address table address
4224                            <1>                      ; Address Table:
4225                            <1>                      ;    Offset 0  : open code address
4226                            <1>                      ;    Offset 4  : read code address
4227                            <1>                      ;    Offset 8  : write code address
4228                            <1>                      ;    Offset 12 : close code address
4229                            <1>                      ;    Offset 16 : IOCTL code address
4230                            <1>                      ;    Offset 20 : initialization code address
4231                            <1>                      ;    Offset 24 : description table address
4232                            <1>                      ;    Offset 28 : configuration table address
4233                            <1>                      ;    Offset 32 : device name address
4234                            <1>                      ;    Offset 36 : dispatch table address
```

```
4235                            <1>                     ;          (for calling kernel-device functions)
4236                            <1>
4237 000195D8 <res 40h>         <1> IDEV_NAME:  resb 8*NUMIDEV
4238                            <1>                     ; 8 byte names of installable device drivers
4239                            <1>
4240 00019618 ????????????????  <1> IDEV_TYPE:  resb NUMIDEV ; Driver type of installable device drivers
4241 00019620 ????????????????  <1> IDEV_FLAGS: resb NUMIDEV ; Device access parameters for installable
4242                            <1>                         ; device drivers (These values are set while
4243                            <1>                          ; the device driver is being loaded.)
4244 00019628 <res 20h>         <1> IDEV_OADDR: resd NUMIDEV ; open function addr for installable dev driver
4245 00019648 <res 20h>         <1> IDEV_CADDR: resd NUMIDEV ; close function addr for installable dev driver
4246 00019668 <res 20h>         <1> IDEV_RADDR: resd NUMIDEV ; read function addr for installable dev driver
4247 00019688 <res 20h>         <1> IDEV_WADDR: resd NUMIDEV ; write function addr for installable dev driver
4248                            <1>
4249                            <1> ; 08/10/2016
4250                            <1> ; 07/10/2016
4251                            <1> ; Device Open and Access parameters
4252 000196A8 <res 1Eh>         <1> DEV_ACCESS:   resb NUMOFDEVICES ; bit 0 = accessable by normal users
4253                            <1>                                 ; bit 1 = read access permission
4254                            <1>                                 ; bit 2 = write access permission
4255                            <1>                                 ; bit 3 = IOCTL permission to users
4256                            <1>                                 ; bit 4 = block device if it is set
4257                            <1>                                 ; bit 5 = 16 bit or 1024 byte data
4258                            <1>                                 ; bit 6 = 32 bit or 2048 byte data
4259                            <1>                                 ; bit 7 = installable device driver
4260 000196C6 <res 1Eh>         <1> DEV_R_OWNER: resb NUMOFDEVICES   ; Reading owner no (u.uid) of devices
4261 000196E4 <res 1Eh>         <1> DEV_R_OPENCOUNT: resb NUMOFDEVICES ; Reading open count
4262 00019702 <res 1Eh>         <1> DEV_W_OWNER: resb NUMOFDEVICES   ; Writing owner no (u.uid) of devices
4263 00019720 <res 1Eh>         <1> DEV_W_OPENCOUNT: resb NUMOFDEVICES ; Writing open count
4264 0001973E <res 1Eh>         <1> DEV_DRIVER:   resb NUMOFDEVICES   ; device driver number (1 to 7Fh)
4265                            <1>                                 ; *if bit 7 is set (80 to FFh)
4266                            <1>                                 ; *if it is installable device driver
4267                            <1>                                 ; *index (0 to 7Fh)
4268                            <1>                                 ; otherwise it is kernel device index
4269 0001975C <res 1Eh>         <1> DEV_OPENMODE:  resb NUMOFDEVICES   ; 1 = read mode
4270                            <1>                                 ; 2 = write mode
4271                            <1>                                 ; 3 = read & write
4272                            <1>                                 ; 0 = not open (free)
4273 0001977A <res 78h>         <1> DEV_NAME_PTR:  resd NUMOFDEVICES   ; pointers to name addresses of drivers
4274                            <1>                                 ; Address base: KDEV_NAME+
4275                            <1>                                 ; or IDEV_NAME+
4276 000197F2 <res 78h>         <1> DEV_R_POINTER: resd NUMOFDEVICES   ; reading pointer, writing pointer
4277 0001986A <res 78h>         <1> DEV_W_POINTER: resd NUMOFDEVICES   ; sector number if block device
4278                            <1>                                 ; character offset if char device
4279 000198E2 ????              <1> alignb 4
4280                            <1>
4281                            <1> ; 06/10/2016
4282                            <1> ; Open File Parameters
4283 000198E4 <res 28h>         <1> OF_FCLUSTER: resd OPENFILES  ; First clusters of open files
4284 0001990C <res Ah>          <1> OF_DRIVE:    resb OPENFILES  ; Logical DOS drive numbers of open files
4285 00019916 <res Ah>          <1> OF_MODE:     resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
4286 00019920 <res Ah>          <1> OF_STATUS:   resb OPENFILES  ; (bit 0 = read, bit 1 = write)
4287 0001992A <res Ah>          <1> OF_OPENCOUNT: resb OPENFILES  ; Open counts of open files
4288 00019934 <res 28h>         <1> OF_POINTER:  resd OPENFILES   ; File seek/read/write pointer
4289 0001995C <res 28h>         <1> OF_SIZE:     resd OPENFILES   ; File sizes of open files (in bytes)
4290 00019984 <res 28h>         <1> OF_DIRFCLUSTER: resd OPENFILES ; Directory First Clusters of open files
4291 000199AC <res 28h>         <1> OF_DIRCLUSTER: resd OPENFILES ; Directory (Entry) Clusters of open files
4292 000199D4 <res 28h>         <1> OF_VOLUMEID: resd OPENFILES  ; Vol ID for removable drives of open files
4293 000199FC <res 28h>         <1> OF_CCLUSTER: resd OPENFILES  ; Current clusters of open files
4294 00019A24 <res 28h>         <1> OF_CCINDEX:  resd OPENFILES  ; Cluster index numbers of current clusters
4295                            <1> ; 24/10/2016
4296 00019A4C <res 14h>         <1> OF_DIRENTRY: resw OPENFILES  ; Directory entry index no. in dir cluster
4297                            <1>                              ; Sector index = entry index / 16
4298                            <1> ;alignb 2
4299                            <1>
4300 00019A60 <res 60h>         <1> DTA:         resd 24      ; Find First File data transfer area
4301                            <1>
4302                            <1> ; 19/12/2016
4303 00019AC0 ??                <1> tcallback:   resb 1       ; Timer callback method flag for 'systimer'
4304 00019AC1 ??                <1> trtc:        resb 1       ; Timer interrupt type flag for 'systimer'
4305                            <1> ; 20/02/2017
4306 00019AC2 ??                <1> no_page_swap: resb 1      ; Swap lock for Signal Response Byte pages
4307                            <1> ;;15/01/2017
4308                            <1> ; 02/01/2017
4309                            <1> ;;intflg:    resb 1       ; software interrupt in progress signal
4310                            <1>                           ; (for timer interrupt)
4311                            <1>
4312 00019AC3 ??                <1> alignb 4
4313                            <1> ; 13/04/2017
4314 00019AC4 <res 1Eh>         <1> DEV_INTR:    resb NUMOFDEVICES ; Device Interrupt (IRQ) number + 1
4315                            <1>                              ; (0= not available, 1= IRQ 0, 16= IRQ 15)
4316 00019AE2 <res 40h>         <1> DEV_INT_HNDLR: resd 16    ; Device Interrupt Handler addr, if > 0
4317                            <1>
4318                            <1>
4319                            <1> ;alignb 4
4320                            <1>
4321                            <1> ; 26/02/2017 ; IRQ Callback parameters ('syscalbac')
4322                            <1> ;Index: ; 0 to 8
4323                            <1> ;     0 = IRQ3, 1 = IRQ4, 2 = IRQ5, 3 = IRQ7
4324                            <1> ;     4 = IRQ9, 5 = IRQ10, 6 = IRQ11, 7 = IRQ12, 8 = IRQ13
4325 00019B22 <res 9h>          <1> IRQ.owner:   resb 9       ; owner, 0 = free, >0 = [u.uno]
4326 00019B2B <res 9h>          <1> IRQ.dev:     resb 9       ; 0 = default/kernel, >0 = device number
4327 00019B34 <res 9h>          <1> IRQ.method:  resb 9       ; 0 = Signal Response Byte, 1 = Callback
4328 00019B3D <res 9h>          <1> IRQ.srb:     resb 9       ; Signal Response/Return Byte value
4329 00019B46 <res 24h>         <1> IRQ.addr:    resd 9       ; Rignal Response Byte address (physical)
4330                            <1>                           ; or Callback service address (virtual)
4331                            <1> ; 28/02/2017
4332 00019B6A ????????          <1> IRQ_cr3:     resd 1       ; for saving cr3 register in IRQ handler
4333 00019B6E ??                <1> IRQnum:      resb 1       ; IRQ number for IRQ handler (trdosk8.s)
4334                            <1>
4335                            <1> ; 10/04/2017
4336                            <1> ; 03/04/2017
4337                            <1> ; UNINITIALIZED AUDIO DATA
4338 00019B6F ??                <1> alignb 4
4339 00019B70 ??                <1> audio_pci: resb 1
```

```
4340 00019B71 ??                <1> audio_device: resb 1
4341 00019B72 ??                <1> audio_mode:  resb 1
4342 00019B73 ??                <1> audio_intr:  resb 1
4343 00019B74 ??                <1> audio_busy:  resb 1  ; Busy flag for audio irq ; 21/04/2017
4344 00019B75 ??                <1> audio_reserved: resb 1
4345 00019B76 ????              <1> audio_io_base:   resw 1   ; Base I/O address of audio device
4346 00019B78 ????????          <1> audio_dev_id:    resd 1   ; BUS/DEV/FN ; 00000000BBBBBBBBDDDDDFFF00000000
4347 00019B7C ????????          <1> audio_vendor:    resd 1
4348 00019B80 ????????          <1> audio_stats_cmd: resd 1
4349                            <1> ;
4350 00019B84 ????????          <1> audio_buffer:   resd 1  ; virtual address of user's audio buffer
4351 00019B88 ????????          <1> audio_p_buffer: resd 1  ; Physical address of user's audio buffer
4352 00019B8C ????????          <1> audio_buff_size: resd 1 ; user's audio buffer size (half buffer size)
4353 00019B90 ????????          <1> audio_dma_buff: resd 1  ; dma buffer address
4354 00019B94 ????????          <1> audio_dmabuff_size: resd 1 ; dma buffer size (2 * half buffer size)
4355 00019B98 ??                <1> audio_flag:  resb 1  ; dma buffer flag (1st half = 0, 2nd half = 1)
4356 00019B99 ??                <1> audio_user:  resb 1 ; user number of the owner
4357 00019B9A ??                <1> audio_cb_mode:     resb 1 ; 0 = signal response byte method
4358                            <1>                    ; 1 = callback method
4359                            <1>                    ; 2 = s.r.b. method with auto increment
4360 00019B9B ??                <1> audio_srb:   resb 1 ; signal response byte value
4361 00019B9C ????????          <1> audio_cb_addr:     resd 1  ; callback service address or s.r.b. address
4362                            <1>                    ; (s.r.b. addr is physical, cbs addr is virtual)
4363                            <1>
4364 00019BA0 ??                <1> audio_bps:   resb 1  ; selected mode: 8 bit, 16 bit
4365 00019BA1 ??                <1> audio_stmo:  resb 1 ; selected mode: mono /stereo
4366 00019BA2 ????              <1> audio_freq:  resw 1 ; sampling rate
4367                            <1>
4368                            <1> ; 21/04/2017
4369 00019BA4 ??                <1> audio_play_cmd: resb 1  ; Play/Stop command (1 = play, 0 = stop)
4370                            <1> audio_civ: ; 28/05/2017 ; Current Buffer Index (AC'97)
4371 00019BA5 ??                <1> audio_flag_eol:    resb 1  ; End of Link status (vt8233, EOL/FLAG)
4372                            <1>
4373                            <1> audio_master_volume:
4374 00019BA6 ??                <1> audio_master_volume_l: resb 1 ; sound volume (lineout) left channel
4375 00019BA7 ??                <1> audio_master_volume_r: resb 1 ; sound volume (lineout) right channel
4376                            <1>
4377                            <1> alignb 4
4378                            <1> ; 28/05/2017
4379                            <1> ; AC'97 Audio Controller Base Adress Registers
4380 00019BA8 ????              <1> NAMBAR:            resw 1 ; Native Audio Mixer Base Address
4381 00019BAA ????              <1> NABMBAR:    resw 1 ; Native Audio Bus Mastering Base Address
4382                            <1>
4383                            <1> ;alignb 4
4384                            <1> ; 21/04/2017
4385 00019BAC <res 400h>        <1> audio_bdl_buff:    resd 32*8  ; VT8233 (AC97) BDL Buffer Size
4386                            <1> ; 12/05/2017
4387 00019FAC ????????          <1> base_addr:  resd 1 ; 'direct_memory_access' (memory.s)
4388                            <1>
4389                            <1> ; 28/08/2017
4390                            <1> ; 20/08/2017
4391 00019FB0 ??                <1>             resb 1  ;
4392 00019FB1 ??                <1> dma_user:   resb 1 ; user number for sysdma
4393 00019FB2 ??                <1> dma_channel: resb 1 ; dma channel for sysdma
4394 00019FB3 ??                <1> dma_mode:   resb 1  ; dma mode for sysdma
4395 00019FB4 ????????          <1> dma_addr:   resd 1 ; dma buffer physical addr for sysdma
4396 00019FB8 ????????          <1> dma_size:   resd 1  ; dma buffer size (in bytes) for sysdma
4397 00019FBC ????????          <1> dma_start:  resd 1  ; dma start address for sysdma
4398 00019FC0 ????????          <1> dma_count:  resd 1  ; dma count (in bytes) for sysdma
4399                            <1>
4400 00019FC4 <res 603Ch>       <1> alignb 65536
4401                            <1> ; 09/08/2017
4402                            <1> ; 12/05/2017
4403 00020000 <res 10000h>      <1> sb16_dma_buffer: resb 65536 ; DMA buffer for sb16 audio playing.
3644                                ; 24/01/2016
3645                                %include 'ubss.s'  ; UNINITIALIZED KERNEL (USER) DATA
3646                            <1> ; ****************************************************************************
3647                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED USER DATA : ubss.s
3648                            <1> ; ----------------------------------------------------------------------------
3649                            <1> ; Last Update: 28/02/2017
3650                            <1> ; ----------------------------------------------------------------------------
3651                            <1> ; Beginning: 24/01/2016
3652                            <1> ; ----------------------------------------------------------------------------
3653                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3654                            <1> ; ----------------------------------------------------------------------------
3655                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3656                            <1> ; ux.s (04/12/2015)
3657                            <1> ; ****************************************************************************
3658                            <1>
3659                            <1> ; Retro UNIX 386 v1 Kernel - ux.s
3660                            <1> ; Last Modification: 04/12/2015
3661                            <1> ;
3662                            <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
3663                            <1> ; (Modified from
3664                            <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
3665                            <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
3666                            <1> ; ----------------------------------------------------------------------------
3667                            <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
3668                            <1> ; (Original) Source Code by Ken Thompson (1971-1972)
3669                            <1> ; <Bell Laboratories (17/3/1972)>
3670                            <1> ; <Preliminary Release of UNIX Implementation Document>
3671                            <1> ; (Section E10 (17/3/1972) - ux.s)
3672                            <1> ; ****************************************************************************
3673                            <1>
3674                            <1> alignb 2
3675                            <1>
3676                            <1> inode:
3677                            <1>       ; 11/03/2013.
3678                            <1>       ;Derived from UNIX v1 source code 'inode' structure (ux).
3679                            <1>       ;i.
3680                            <1>
3681 00030000 ????              <1>       i.flgs: resw 1
3682 00030002 ??                <1>       i.nlks: resb 1
3683 00030003 ??                <1>       i.uid:  resb 1
3684                            <1>       ;i.size: resw 1 ; size
```

```
3685 00030004 ????              <1>       resw 1 ; 29/04/2016
3686 00030006 <res 10h>         <1>       i.dskp: resw 8 ; 16 bytes
3687 00030016 ????????          <1>       i.ctim: resd 1
3688 0003001A ????????          <1>       i.mtim: resd 1
3689 0003001E ????              <1>       i.rsvd: resw 1 ; Reserved (ZERO/Undefined word for UNIX v1.)
3690                            <1>
3691                            <1> I_SIZE      equ $ - inode
3692                            <1>
3693                            <1> process:
3694                            <1>       ; 19/12/2016
3695                            <1>       ; 21/05/2016
3696                            <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
3697                            <1>       ; 06/05/2015 - Retro UNIX 386 v1
3698                            <1>       ; 11/03/2013 - 05/02/2014 (Retro UNIX 8086 v1)
3699                            <1>       ;Derived from UNIX v1 source code 'proc' structure (ux).
3700                            <1>       ;p.
3701                            <1>
3702 00030020 <res 20h>         <1>       p.pid:   resw nproc
3703 00030040 <res 20h>         <1>       p.ppid:  resw nproc
3704 00030060 <res 20h>         <1>       p.break: resw nproc
3705 00030080 <res 10h>         <1>       p.ttyc:  resb nproc ; console tty in Retro UNIX 8086 v1.
3706 00030090 <res 10h>         <1>       p.waitc: resb nproc ; waiting channel in Retro UNIX 8086 v1.
3707 000300A0 <res 10h>         <1>       p.link: resb nproc
3708 000300B0 <res 10h>         <1>       p.stat: resb nproc
3709                            <1>
3710                            <1>       ; 06/05/2015 (Retro UNIX 386 v1 feature only !)
3711 000300C0 <res 40h>         <1>       p.upage: resd nproc ; Physical address of the process's
3712                            <1>                           ; 'user' structure
3713                            <1>       ; 21/05/2016
3714                            <1>       ; 19/05/2016 (TRDOS 386 feature only!)
3715 00030100 <res 10h>         <1>       p.timer: resb nproc ; number of timer events of the processs
3716                            <1>
3717                            <1>       ; 19/12/2016
3718 00030110 <res 40h>         <1>       p.tcb: resd nproc ; timer callback service address (if > 0)
3719                            <1>
3720                            <1> P_SIZE equ $ - process
3721                            <1>
3722                            <1> ; fsp table (original UNIX v1)
3723                            <1> ;
3724                            <1> ;Entry
3725                            <1> ;          15                                      0
3726                            <1> ;  1     |---|--------------------------------------|
3727                            <1> ;        |r/w|       i-number of open file          |
3728                            <1> ;        |---|--------------------------------------|
3729                            <1> ;        |               device number              |
3730                            <1> ;        |------------------------------------------|
3731                            <1> ;    (*) | offset pointer, i.e., r/w pointer to file |
3732                            <1> ;        |------------------------------------------|
3733                            <1> ;        |  flag that says    | number of processes  |
3734                            <1> ;        |   file deleted     | that have file open  |
3735                            <1> ;        |------------------------------------------|
3736                            <1> ;  2     |                                          |
3737                            <1> ;        |------------------------------------------|
3738                            <1> ;        |                                          |
3739                            <1> ;        |------------------------------------------|
3740                            <1> ;        |                                          |
3741                            <1> ;        |------------------------------------------|
3742                            <1> ;        |                                          |
3743                            <1> ;        |------------------------------------------|
3744                            <1> ;  3     |                                          |
3745                            <1> ;        |                                          |
3746                            <1> ;
3747                            <1> ; (*) Retro UNIX 386 v1 modification: 32 bit offset pointer
3748                            <1>
3749                            <1>
3750                            <1> ; 15/04/2015
3751 00030150 <res 1F4h>        <1> fsp:   resb nfiles * 10 ; 11/05/2015 (8 -> 10)
3752 00030344 ????              <1> idev:  resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
3753 00030346 ????              <1> cdev:  resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
3754                            <1> ; 18/05/2015
3755                            <1> ; 26/04/2013 device/drive parameters (Retro UNIX 8086 v1 feature only!)
3756                            <1> ; 'UNIX' device numbers (as in 'cdev' and 'u.cdrv')
3757                            <1> ;     0 -> root device (which has Retro UNIX 8086 v1 file system)
3758                            <1> ;     1 -> mounted device (which has Retro UNIX 8086 v1 file system)
3759                            <1> ; 'Retro UNIX 8086 v1' device numbers: (for disk I/O procedures)
3760                            <1> ;     0 -> fd0 (physical drive, floppy disk 1), physical drive number = 0
3761                            <1> ;     1 -> fd1 (physical drive, floppy disk 2), physical drive number = 1
3762                            <1> ;     2 -> hd0 (physical drive, hard disk 1), physical drive number = 80h
3763                            <1> ;     3 -> hd1 (physical drive, hard disk 2), physical drive number = 81h
3764                            <1> ;     4 -> hd2 (physical drive, hard disk 3), physical drive number = 82h
3765                            <1> ;     5 -> hd3 (physical drive, hard disk 4), physical drive number = 83h
3766 00030348 ??                <1> rdev:  resb 1 ; root device number ; Retro UNIX 8086 v1 feature only!
3767                            <1>               ; as above, for physical drives numbers in following table
3768 00030349 ??                <1> mdev:  resb 1 ; mounted device number ; Retro UNIX 8086 v1 feature only!
3769                            <1> ; 15/04/2015
3770 0003034A ??                <1> active:      resb 1
3771 0003034B ??                <1>       resb 1 ; 09/06/2015
3772 0003034C ????              <1> mnti:  resw 1
3773 0003034E ????              <1> mpid:  resw 1
3774 00030350 ????              <1> rootdir: resw 1
3775                            <1>
3776                            <1> ; 21/05/2016 - TRDOS 386 (TRDOS v2.0) - priority levels, 3 run queues
3777                            <1> runq:
3778 00030352 ????              <1> runq_event:   resw 1 ; high priority, 'run for event'            ; 2
3779 00030354 ????              <1> runq_normal:  resw 1 ; normal/regular priority, 'run as requlat' ; 1
3780 00030356 ????              <1> runq_background: resw 1 ; low priority, 'run on background'         ; 0
3781                            <1> ;
3782 00030358 ??                <1> imod:  resb 1
3783 00030359 ??                <1> smod:  resb 1
3784 0003035A ??                <1> mmod:  resb 1
3785 0003035B ??                <1> sysflg:  resb 1
3786                            <1>
3787                            <1> alignb 4
3788                            <1>
3789                            <1> user:
```

```
                                        <1>       ; 13/01/2017
                                        <1>       ; 19/12/2016
                                        <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
                                        <1>       ;            [u.pri] usage method modification
                                        <1>       ; 04/12/2015
                                        <1>       ; 18/10/2015
                                        <1>       ; 12/10/2015
                                        <1>       ; 21/09/2015
                                        <1>       ; 24/07/2015
                                        <1>       ; 16/06/2015
                                        <1>       ; 09/06/2015
                                        <1>       ; 11/05/2015
                                        <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit modifications)
                                        <1>       ; 10/10/2013
                                        <1>       ; 11/03/2013.
                                        <1>       ;Derived from UNIX v1 source code 'user' structure (ux).
                                        <1>       ;u.
                                        <1>
0003035C ????????                       <1>       u.sp:    resd 1 ; esp (kernel stack at the beginning of 'sysent')
00030360 ????????                       <1>       u.usp:   resd 1 ; esp (kernel stack points to user's registers)
00030364 ????????                       <1>       u.r0:    resd 1 ; eax
00030368 ????                           <1>       u.cdir:   resw 1
0003036A <res Ah>                       <1>       u.fp:    resb 10
00030374 ????????                       <1>       u.fofp:   resd 1
00030378 ????????                       <1>       u.dirp:   resd 1
0003037C ????????                       <1>       u.namep:  resd 1
00030380 ????????                       <1>       u.off:   resd 1
00030384 ????????                       <1>       u.base:   resd 1
00030388 ????????                       <1>       u.count:  resd 1
0003038C ????????                       <1>       u.nread:  resd 1
00030390 ????????                       <1>       u.break:  resd 1 ; break
00030394 ????                           <1>       u.ttyp:   resw 1
                                        <1>       ; 10/01/2017 (TRDOS 386, relocation and dword alignment)
                                        <1>       ; tty number (rtty, rcvt, wtty)
00030396 ??                             <1>       u.ttyn:  resb 1 ; 28/07/2013 - Retro Unix 8086 v1 feature only !
00030397 ??                             <1>       u.resb:   resb 1 ; 10/01/2017 (TRDOS 386, temporary)
00030398 <res 10h>                      <1>       u.dirbuf: resb 16 ; 04/12/2015 (10 -> 16)
                                        <1>       ;u.pri:  resw 1 ; 14/02/2014
000303A8 ??                             <1>       u.quant:  resb 1 ; Retro UNIX 8086 v1 Feature only ! (uquant)
000303A9 ??                             <1>       u.pri:   resb 1 ; Modification: 21/05/2016 (priority levels: 0, 1, 2)
000303AA ????                           <1>       u.intr:   resw 1
000303AC ????                           <1>       u.quit:   resw 1
                                        <1>       ;u.emt:   resw 1 ; 10/10/2013
                                        <1>       ;u.ilgins: resw 1 ; 10/01/2017
000303AE ????                           <1>       u.cdrv:   resw 1 ; cdev
000303B0 ??                             <1>       u.uid:   resb 1 ; uid
000303B1 ??                             <1>       u.ruid:   resb 1
000303B2 ??                             <1>       u.bsys:   resb 1
000303B3 ??                             <1>       u.uno:   resb 1
000303B4 ????????                       <1>       u.upage:  resd 1 ; 16/04/2015 - Retro Unix 386 v1 feature only !
000303B8 ????????                       <1>       u.pgdir:  resd 1 ; 09/03/2015 (page dir addr of process)
000303BC ????????                       <1>       u.ppgdir: resd 1 ; 06/05/2015 (page dir addr of the parent process)
000303C0 ????????                       <1>       u.pbase:  resd 1 ; 20/05/2015 (physical base/transfer address)
000303C4 ????                           <1>       u.pcount: resw 1 ; 20/05/2015 (byte -transfer- count for page)
                                        <1>       ;u.pncount: resw 1
                                        <1>             ; 16/06/2015 (byte -transfer- count for page, 'namei', 'mkdir')
                                        <1>       ;u.pnbase:  resd 1
                                        <1>             ; 16/06/2015 (physical base/transfer address, 'namei', 'mkdir')
                                        <1>                     ; 09/06/2015
000303C6 ??                             <1>       u.kcall:  resb 1 ; The caller is 'namei' (dskr) or 'mkdir' (dskw) sign
000303C7 ??                             <1>       u.brwdev: resb 1 ; Block device number for direct I/O (bread & bwrite)
                                        <1>                     ; 24/07/2015 - 24/06/2015
                                        <1>       ;u.args:  resd 1 ; arguments list (line) offset from start of [u.upage]
                                        <1>                     ; (arg list/line is from offset [u.args] to 4096 in [u.upage])
                                        <1>                     ; ([u.args] points to argument count -argc- address offset)
                                        <1>                     ; 24/06/2015
                                        <1>       ;u.core:  resd 1 ; physical start address of user's memory space (for sys exec)
                                        <1>       ;u.ecore: resd 1 ; physical end address of user's memory space (for sys exec)
                                        <1>       ; last error number
000303C8 ????????                       <1>       u.error:  resd 1 ; 28/07/2013 - 09/03/2015
                                        <1>                      ; Retro UNIX 8086/386 v1 feature only!
                                        <1>                     ; 21/09/2015 (debugging - page fault analyze)
000303CC ????????                       <1>       u.pfcount: resd 1 ; page fault count for (this) process (for sys geterr)
                                        <1>             ; 19/12/2016 (TRDOS 386)
000303D0 ????????                       <1>       u.tcb:   resd 1 ; Timer callback address/flag which will be used by timer int
                                        <1>             ; 13/01/2017 (TRDOS 386)
000303D4 ??                             <1>       u.t_lock: resb 1 ; Timer interrupt (callback) lock (unlocked by 'sysrele')
000303D5 ??                             <1>       u.t_mode: resb 1 ; running mode during timer interrupt (0= system, 0FFh= user)
                                        <1>             ; 26/02/2017 (TRDOS 386)
000303D6 ??                             <1>       u.irqc:  resb 1  ; Count of IRQ callback services (IRQs in use)
                                        <1>             ; 28/02/2017 (TRDOS 386)
000303D7 ??                             <1>       u.irqwait: resb 1 ; IRQ waiting for callback service flag (IRQ number, If > 0)
000303D8 ??                             <1>       u.r_lock:  resb 1 ; 'IRQ callback service is in progress' flag (IRQ lock)
000303D9 ??                             <1>       u.r_mode:  resb 1 ; running mode during hadware interrupt
                                        <1>             ; 27/02/2017 (TRDOS 386)
000303DA ??                             <1>       u.fpsave: resb 1  ; TRDOS 386, 'save/restore FPU registers' flag
000303DB ??                             <1> alignb 4
000303DC <res 5Eh>                      <1>       u.fpregs: resb 94 ; 94 byte area for saving and restoring FPU registers
                                        <1>
0003043A ????                           <1> alignb 4
                                        <1>
                                        <1> U_SIZE equ $ - user
                                        <1>
                                        <1> ; 18/10/2015 - Retro UNIX 386 v1 (local variables for 'namei' and 'sysexec')
0003043C ????????                       <1> pcore:  resd 1  ; physical start address of user's memory space (for sys exec)
00030440 ????????                       <1> ecore:  resd 1  ; physical address of user's stack/last page (for sys exec)
00030444 ????????                       <1> nbase:  resd 1  ; physical base address for 'namei' & 'sysexec'
00030448 ????                           <1> ncount: resw 1  ; remain byte count in page for 'namei' & 'sysexec'
0003044A ????                           <1> argc: resw 1   ; argument count for 'sysexec'
0003044C ????????                       <1> argv: resd 1   ; argument list (recent) address for 'sysexec'
                                        <1>
                                        <1> ; 03/06/2015 - Retro UNIX 386 v1 Beginning
                                        <1> ; 07/04/2013 - 31/07/2013 - Retro UNIX 8086 v1
00030450 ??                             <1> rw:    resb 1 ;; Read/Write sign (iget)
                                        <1>
```

```
                            <1> ;alignb 4
                            <1>
                            <1> ; 24/04/2016
00030451 ????????           <1> ii:         resd 1 ; first cluster of the program file
00030455 ????????           <1> i.size:     resd 1 ; size of the program file

00030459 ??????                 alignb 4

                                ; 23/05/2016 (TRDOS 386)
                                ; 14/10/2015 (Retro UNIX 386 v1, 'unix386.s')
0003045C ????????               cr3reg:      resd 1  ; cr3 register content at the beginning of the timer
                                             ; (or RTC) interrupt handler.

                                ; 10/12/2016 (callback)
                                ; 10/06/2016
                                ; 19/05/2016
                                ; 18/05/2016 - TRDOS 386 feature only !
00030460 <res 100h>             timer_set: resd 16*4   ; 256 bytes memory space for 16 timer events
                                      ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
                                      ;      Owner:       resb 1 ; 0 = free
                                      ;                           ;>0 = process number (u.uno)
                                      ;      Callback:    resb 1 ; 0 = response byte address (phy)
                                      ;                           1 = callback address (virtual)
                                      ;      Interrupt:   resb 1 ; 0 = Timer interrupt (or none)
                                      ;                           ; 1 = Real Time Clock interrupt
                                      ;      Response:    resb 1 ; 0 to 255, signal return value
                                      ;      Count Limit: resd 1 ; count of ticks (total/set)
                                      ;      Current Count: resd 1 ; count of ticks (current)
                                      ;      Response Addr: resd 1 ; response byte (pointer) address
                                      ;                           ; (or callback -user service- address)

                                ;; Memory (swap) Data (11/03/2015)
                                ; 09/03/2015
00030560 ????                   swpq_count: resw 1 ; count of pages on the swap queue
00030562 ????????               swp_drv:    resd 1 ; logical drive description table address of the swap drive/disk
00030566 ????????               swpd_size:  resd 1 ; size of swap drive/disk (volume) in sectors (512 bytes).

0003056A ????????               swpd_free:  resd 1 ; free page blocks (4096 bytes) on swap disk/drive (logical)
0003056E ????????               swpd_next:  resd 1 ; next free page block
00030572 ????????               swpd_last:  resd 1 ; last swap page block

00030576 ????                   alignb 4

                                ; 10/07/2015
                                ; 28/08/2014
00030578 ????????               error_code: resd 1
                                ; 29/08/2014
0003057C ????????               FaultOffset:       resd 1
                                ; 21/09/2015
00030580 ????????               PF_Count:   resd 1 ; total page fault count
                                                   ; (for debugging - page fault analyze)
                                                   ; 'page_fault_handler' (memory.inc)
                                                   ; 'sysgeterr' (u9.s)

                                ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
                                ; 22/08/2015 (Retro UNIX 386 v1)
                                buffer:
00030584 ????????????????            resb   8
                                readi_buffer:
0003058C <res 200h>                   resb   512
0003078C ????????????????            resb   8
                                writei_buffer:
00030794 <res 200h>                   resb   512
                                ; 24/10/2016
00030994 ????????????????            resb   8
                                rw_buffer:
0003099C <res 800h>                   resb   2048  ; general purposed, r/w sector buffer

                                %if 1
                                ; 17/01/2021
0003119C <res 80h>              edid_info:  resb 128 ; VESA EDID (monitor capabilities) info
                                ; 28/11/2020
0003121C ??                     pmi32:      resb 1 ; (>0) use VESA VBE3 protected mode calls
0003121D ??                     vbe_mode_x: resb 1 ; VESA VBE3 video bios mode set options
0003121E ????                   video_mode: resw 1 ; VESA VBE3 video mode (with option flags)
                                ; 30/11/2020
00031220 ????????               vbe3bios_addr:     resd 1 ; new (writable mem) address of VBE3 bios
                                ; 02/12/2020
00031224 ????????               pmid_addr:  resd 1 ; PMInfoBlock ('PMID') linear address
                                ; 14/01/2021
                                ; 06/12/2020       ; VESA VBE 3 video state
                                ;vbe3stbufsize: resw 1    ; video regs/dac/bios state buffer size
                                ;                  ; block size in bytes
                                ; 16/01/2021
00031228 ????                   vbe3stbsflags: resw 1 ; video regs/dac/bios state buffer size
                                ;                  ; pointer flags for buffer state options
                                %endif

                                %if 1
                                ; 10/12/2020
                                LFB_Info:
0003122A <res 10h>                          resb 16 ; Linear Frame Buffer info block

                                ;24/11/2020 - TRDOS 386 v2.0.3
                                ; BOCHS/PLEX86 VESA VBE3 MODE INFO extension to TRDOS 386 v2 kernel
                                MODE_INFO_LIST:
0003123A <res 44h>                          resb 68 ; mode + 66 byte VESA vbe3 mode info (4F01h)
                                %endif

                                ; 05/01/2021
0003127E ??                     ufont:      resb 1 ; (VGA graphics) user font flags
                                                   ; bit 7 - permission flag for int 31h
                                                   ; bit 4 - 8x16 user font ready/loaded flag
                                                   ; bit 3 - 8x8 user font ready/loaded flag
                                                   ; bit 1 - select 8x16 user font (sysvideo)
```

```
3745                                            ; bit 0 - select 8x8 user font (sysvideo)
3746 0003127F ??                           resb 1 ; 19/01/2021
3747                                  ; 17/01/2021
3748 00031280 ??                      srvsf:      resb 1 ; 'save restore video state' permission flag
3749                                  ; 18/01/2021
3750 00031281 ??                      srvso:      resb 1 ; video state buffer save/restore option
3751 00031282 ????????                VideoStateID: resd 1 ; used to verify state saved by same prog
3752                                  ; 29/01/2021
3753 00031286 ????                    v_width:    resw 1 ; screen (display page) width
3754 00031288 ??                      v_ops:      resb 1 ; 'sysvideo' graphics data transfer option
3755 00031289 ??                      v_bpp:      resb 1 ; bits per pixels ('sysvideo')
3756 0003128A ????????                v_mem:      resd 1 ; video memory ('sysvideo')
3757 0003128E ????????                v_siz:      resd 1 ; video page size ('sysvideo')
3758 00031292 ????????                v_str:      resd 1 ; window start adress ('sysvideo')
3759 00031296 ????????                v_end:      resd 1 ; window end (end+1) adress ('sysvideo')
3760                                  ; 31/01/2021
3761                                  ; 01/01/2021
3762                                  ;maskbuff:     ;resd 1    ; user's bitmask buffer addr ('sysvideo')
3763 0003129A ????????                maskcolor:  resd 1 ; VGA/SVGA pixel mask color ('sysvideo')
3764                                  ; 27/02/2021
3765 0003129E ????????                pixcount:   resd 1 ; pixel count ('sysvideo' window ops)
3766                                  ; 02/02/2021
3767 000312A2 ????????????????        buffer8:    resd 2 ; 8 bytes small buffer for 'sysvideo'
3768
3769                                  bss_end:
3770
3771                                  ; 27/12/2013
3772                                  _end:  ; end of kernel code
```